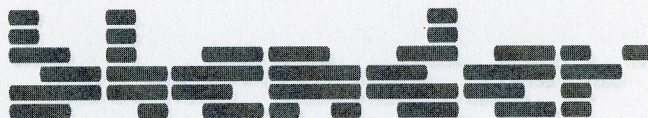




V1.5 manual

Ton Roosendaal





V1.5 manual

Ton Roosendaal







V1 5 manua

Ton Roosendaal



Published by Not a Number
Eindhoven, the Netherlands, 1998

Copyright © 1998 by Not a Number.
All rights reserved.

No part of this book may be reproduced or
transmitted in any form or by any means, electronic
or mechanical, including photocopy, recording, or
any information storage and retrieval system, without
permission in writing from the publisher.

Not a Number

Horsten1
5612 AX Eindhoven
the Netherlands.
blender@blender.nl

Design & Artwork:

Riff Raff, Amsterdam, The Netherlands

DTP:

BeeldBuijs KamelÉon, buijs@bigfoot.com

English translation:

Kenneth Kuhn and Lester Nunnelee

Printing and lithography:

PrePart Digital Printing and Imaging.

Roosendaal, Ton

Blender
V1.5 manual

Eindhoven: Not a Number
ISBN 90-76519-01-3
NUGI 854

Preface

It is now nearly six months since we released Blender on the Internet. Only in my wildest dreams could I have envisioned what would ensue.

From thousands of users, suddenly enriched by this amazing gift that opened the unimaginable world of 3D animation to them. From students by the hundreds in far-away universities, who were able to bring their stuffy Silicons to life again. From professional users, who almost apologetically reported that they were using Blender.

But I had no idea that the stream of reactions would turn out to be so unbelievably satisfying. Reactions from anonymous users, who took the time to write thanking us. From people who wrote saying "I love you", "You guys are gods!", and "You have changed my life forever!".

Being able to make so many people happy at once was addictive. It gave me an irrepressible motivation to keep going - and primarily to keep such reactions coming in. What a pity that the world economy is not based on appreciation!

For during the months since Blender was launched, NeoGeo, the animation studio that gave birth to Blender, decided to go out of business. I had devoted heart and soul to this company for nine years. My friends were there, and suddenly, there was no future there. The departure of NeoGeo ran parallel to the blooming of Blender.

This is my personal motivation for writing this manual. As an expression of gratitude and as a tangible monument to these nine years of my life. It is my thanks to the people at NeoGeo, through whom Blender came into being. I would like to dedicate this book to them.

First and foremost to my friend and business partner, Frank van Beek. His sharp intelligence and loyal friendship have been invaluable to me. And in terms of Blender, without him it would never have evolved past an amateur's attempt to write a ray tracer on Amiga.

Secondly, and foremost as well, thanks to my friend and co-owner, Joeri Kassenaar. Someone who is blessed with an intelligent and critical perspective on life, and who was still able to bring unbridled enthusiasm to working with me.

I would also like to thank Paul Smits, our first 'real' employee. I watched his talents bloom in our years at NeoGeo.

Thanks also to Rob Haarsma. Our artist and enthusiastic 3D VJ hacker.

I want to thank Anja Vugts, who from the beginning kept our administration running.

Thanks as well to Carlo Storimans and John Drenth, our office managers, who turned a stuffy old villa into the palace in which NeoGeo enjoyed its best years.

I wish to thank all the people who took part in the work and life at NeoGeo: Arjan de Haas, Bea van Sweeden, Huub Vossen, Ivan Lodde, Jan Pieter van Seventer, Jo van Beek, Josette Gevers, Jurgen Zweemer, Marc van Kempen, Marte Elings, Mien van Overbruggen, Pieter Koenen, Sebnem Aydeniz, Sven van Sweeden and Willem-Paul van Overbruggen.

And finally, I wish to thank all NeoGeo's customers. Special thanks go to Jan Binkhorst, Jhein Lohman and Jonathan Ellis, for their faith in our potential.

This is the most difficult part of a foreword. How do you decide who to mention and how do you prevent accidentally leaving out a good friend? Sheesh, your name is missing!

Let me close by thanking you, the reader. Thanks Blender users, for your trust, your affection and your attention. Thanks for the warm welcome to the Global Village.

Ton Roosendaal
Eindhoven, October 1998

Contents

Part 1

I never read a manual!

013 *For those who browse.*

014 *Installation*

Installation for SGI users
Installation for Linux or
FreeBSD i386 Systems

019 *The Object Oriented System*

The Main Data Blocks
Data Structure Example
Screens and Windows

024 *Basic Editing*

The Mouse
The Keyboard
Selecting
Active
Moving / Rotating / Scaling
Objects and EditMode
Materials and Textures
Keys and Animation Curves
Saving and Loading Files

Part 2

Do it yourself

033 *Getting started with 3D*

Blender Startup
3D View Manipulation
View Presets
View Modes
Layers
Adding Objects
Material Properties
Mesh Editing
Save and Load

038 *Coffeecup Tutorial*

046 *Flying Logo Tutorial*

061 *More Tutorials!*

Part 3

Blender

063 *The Data Structure*

"What You See Is What You Want"
"What You Do is What You Get!"
The DataBlocks
Objects and ObData
Objects, ObData and Materials
Creating links with HotKeys
Users
Blender File Format
And where is the Undo Buffer?
The Libraries

078 *The Interface*

Screens
Windows
The Buttons

083 *The Menus*

Toolbox
Pup Menus
Number Menus

Part 4

Working with objects

087 **Objects**

- The Object Block
- Selecting and Activating
- Grab / Rotate / Size
- EditMode
- Hierarchies
- ToolBox

091 **The Mesh**

- The Mesh Block
- EditMode
- Extrude
- Spin
- Assigning Materials
- Other Tools

097 **Curve and Surface**

- The Curve block
- Beziers
- Nurbs
- Poly
- Modeling with Curves
- Beveling
- The Surface Block
- Modeling with Surfaces
- Assigning Materials

104 **Font**

- The Font Block
- Entering Text

105 **Lattice**

- The Lattice Block

106 **Ika and Skeletons**

- The Ika Block
- Working with Ika's
- Skeletons

111 **MetaBall**

- The MetaBall Block
- Modeling with MetaBalls

111b **Camera**

- Modeling with Lattices

Part 5

Animation with objects

113 **Object Ipos**

- Keys and Curves
- Ipo Block
- Making Ipos in the 3DWindow
- The IpoWindow
- Beziers
- IpoCurves
- Draw IpoCurves
- Rotations and Scaling
- IpoKeys

123 **Vertex Keys**

- The Key Block
- Mesh Vertex Keys
- The IpoCurve and VertexKey Lines
- Tips
- Curve and Surface Keys
- Lattice Keys

126 **Special Animation Techniques**

- Motion Paths
- Object Tracking
- Duplicators
- Ika and Skeletons
- Particles

Part 6

Materials, light: render!

- 133 **Materials and Textures**
- Rendering
 - Shading
 - Materials
 - The 'mapping'
 - Textures
 - Animated Materials

- 139 **Lamps and Shadow**
- Light Types
 - Shadow Buffers
 - Selective Lighting
 - Lamp Textures
 - Lighting Tips

- 144 **Rendering and Post-production**
- Video Work
 - The Render Engine
 - Within a Pixel
 - Alpha
 - Memory
 - Graphics File Formats
 - Sequence Editor

Part 7

Special techniques

- 155 **VertexPaint**
- 156 **Rotoscoping**
- 159 **Cartoon Animation and 3D**
- 163 **Import Inventor/VRML Files**
- 164 **Videoscape Files**

Part 8

Reference: the windows and hotkeys

- 171 **Blender windows general introduction**
- The Mouse
 - Window Hotkeys
 - Universal Hotkeys

- 174 **The InfoWindow**
- InfoHeader
 - UserMenu

- 178 **The FileWindow**
- FileHeader
 - FileWindow
 - FileSelect
 - Read Libraries
 - FileManager
 - DataView and DataBrowse

- 185 **The 3DWindow**
- 3DHeader
 - 3DWindow
 - The Mouse
 - Numpad
 - Hotkeys
 - EditMode Hotkeys General
 - EditMode Mesh Hotkeys
 - EditMode Curve Hotkeys
 - EditMode Font Hotkeys
 - EditMode Surface Hotkeys
 - Ika Hotkeys
 - Vertexpaint Hotkeys

- 205 **The ButtonsWindow**

- 207 **The IpoWindow**
- IpoHeader
 - IpoWindow
 - The Mouse
 - The Hotkeys

214	The SequenceWindow SequenceHeader SequenceWindow The Mouse The Hotkeys
220	The OpsWindow OpsHeader OpsWindow The Mouse The Hotkeys
224	The ImageWindow ImageHeader ImageWindow
226	The ImageSelect Window ImageSelectHeader ImageSelectWindow The mouse and Hotkeys
228	The Animation Playback Window

Wood Texture
Marble Texture
Blend Texture
Stucci Texture
Noise Texture

260	The WorldButtons
266	The AnimButtons Anim Effects: Build Anim Effects: Particles

273	The EditButtons EditButtons, General EditButtons, Mesh EditButtons, Curve en Surface EditButtons, Font EditButtons, MetaBall EditButtons, Lattice EditButtons, Ika EditButtons, Camera
-----	---

287	The Vertex Paint Buttons
288	The DisplayButtons

Part 9

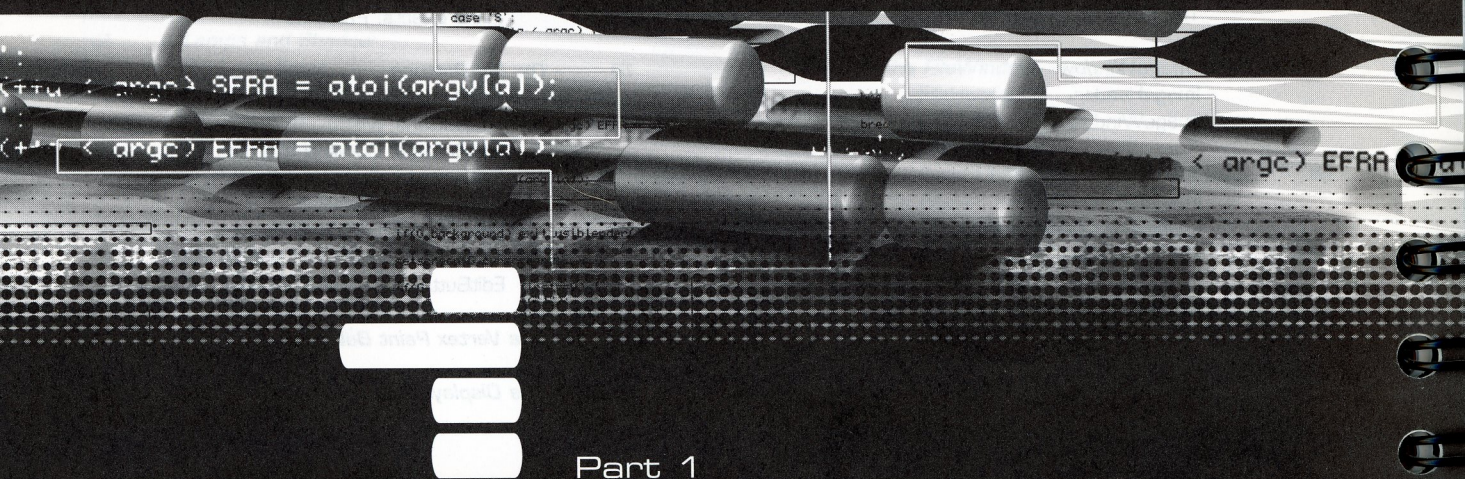
Reference: the Buttons

231	The ViewButtons
233	The LampButtons
238	The MaterialButtons
248	The MaterialButtons, Halos
251	The TextureButtons The standard TextureButtons Colorband Image Texture Plug-in Texture Clouds Texture

Part 10

Appendices

295	Feature Listing
297	Command Line Options
298	Glossary of Blender Terms
302	Index



Part 1

I never read a manual!

For those who browse

This section is for all those people who hate manuals, for those who cannot wait to get started with Blender! Here you find an abstract of the manual. Actually, it is an adaption of the HTML pages that were available at the Blender web site.

If this text is still too theoretical for you, just skip it and continue with the second section: "Do it Yourself". It contains some clear, easy examples that require no understanding of Blender, but will give you some practical experience with the software and, I hope, sufficient motivation to continue learning it.

The next five sections of this manual, sections three to seven, describe all Blender's features arranged by themes. A lot of practical examples are included there as well. The second half of this manual, sections eight and nine, are for reference purposes.

Blender is not an easy application in the usual sense. It has a peculiar interface and a quite a steep learning curve. But once you learn the basics, it will prove to be extremely logical and well-organized.

Happy Blending!

Installation

At the time this manual was written, only three Blender versions were available: For SGI, and for FreeBSD and Linux i386 PC's.

Porting Blender to Linux Alpha, Linux PPC, Sun Solaris and BeOS i386, for example, had not started yet, but these ports will only be released with a full compatibility with the current version, Blender 1.5.

Installation for SGI systems

1. Download

In the Blender download section of the www.blender.nl site you can get the latest Blender release. There are 4 SGI versions available:

- blenderSGI_5.3_iris_1.5x.tar.gz: IRIX 5.3 with IRIS GL
- blenderSGI_6.2_iris_1.5x.tar.gz: IRIX 6.2 with IRIS GL (IRIX 6.x compatible)
- blenderSGI_5.3_ogl_1.5x.tar.gz: IRIX 5.3 with OpenGL
- blenderSGI_6.2_ogl_1.5x.tar.gz: IRIX 6.2 with OpenGL (IRIX 6.x compatible)

For newer SGI systems, e.g. O2 or Octanes, the OpenGL versions are generally faster.

2. Uncompress

Use a UNIX shell to move the download file to the directory in which the Blender installation directory should be created.

Make that directory the current directory. Now type:

```
gunzip blenderSGI_xxx.tar.gz
tar xvf blenderSGI_xxx.tar
```

This creates the directory `./blenderSGI_xxx/`. Initially, it contains the executable and the primary files.

3. Add an environment variable:

To be able to find its data files, Blender needs an environment variable that points to its installation directory, or any other directory you wish to copy the installation files to. The executable itself can be copied to any location you wish.

Important: the `name_of_directory` must be the *full* path, starting with `/` !

```
setenv BLENDERDIR name_of_directory (for csh shell)
export BLENDERDIR=name_of_directory (for sh shell)
```

You may want to enter this into your `.cshrc` or `.login` (or `.profile`) file, so it is automatically set when you login.

4. Start Blender.

Type in the UNIX shell:

```
./blender (if Blender resides in the current directory)
blender (if the PATH environment has been correctly set)
```

Within a few seconds Blender is ready to work with. When you run Blender for the first time, it copies the files `.B.blend` and `.B.fs` to your home directory.

If Blender cannot find the installation files, it terminates with an error message. Carefully check whether the environment variable was set correctly.


```

        break;
    case 'f':
        a++;

```

```
if (G.scene) {
    G.real_sfra = SFRA;
    G.real_efra = EFRA;
}
```

```
/* Signalen zetten. Even wachten met USR1 en INT, want die kunnen * ook het lezen van de file onderbreken. Geeft van (
if (arg0[0] != '\0') {
    signal(SIGUSR2, ignore); /* als er een SIGUSR2 gegeven wordt dan doe de file */ == 'a') {
    signal(SIGINT, ignore);
    signal(SIGUSR1, breakf); /* netje kill gebruikt door (traces)enderdozen */ signal(SIGINT, breakf); /* netje killge
```

```

bzero(&our_sdr, sizeof(struct SDR));
our_sdr.data = &initstr;
initstr = (int *)XDRtoSdr(&our_sdr);
initstr->end = &our_sdr; /* moet vooraan staan */
while (1) {
    if (!background) break;
    a++;
}

```

```
EFRA = SFRA,  
animrender();
```

```
/* fark voordat X opstart (only 50)
a= /* patch zodat glutinit van de glutinit(50)

```

```

goldplayback();
for (int i; i < argo; i++) {
    if (argo[i] == '-') {
        switch (argo[i+1]) {
            case 'p': /* preface */
                ...
            case '4':
                ...
            case '5':
                ...
        }
    }
}

```

0100101 101011010

```
size = atoi(argv[1]);
char *argv[2];
G_weight = fr = SFRA;
```

SFRA = atot tane

100 0101



BY ERIC SPOFFORD

01234567891011121314151617181920212223242526272829303132333435363738394041424344454647484950515253545556575859606162636465666768697071727374757677787980818283848586878889909192939495969798991001011021031041051061071081091101111121131141151161171181191201211221231241251261271281291301311321331341351361371381391401411421431441451461471481491501511521531541551561571581591601611621631641651661671681691701711721731741751761771781791801811821831841851861871881891901911921931941951961971981992002012022032042052062072082092102112122132142152162172182192202212222232242252262272282292302312322332342352362372382392402412422432442452462472482492502512522532542552562572582592602612622632642652662672682692702712722732742752762772782792802812822832842852862872882892902912922932942952962972982993003013023033043053063073083093103113123133143153163173183193203213223233243253263273283293303313323333343353363373383393403413423433443453463473483493503513523533543553563573583593603613623633643653663673683693703713723733743753763773783793803813823833843853863873883893903913923933943953963973983994004014024034044054064074084094104114124134144154164174184194204214224234244254264274284294304314324334344354364374384394404414424434444454464474484494504514524534544554564574584594604614624634644654664674684694704714724734744754764774784794804814824834844854864874884894904914924934944954964974984995005015025035045055065075085095105115125135145155165175185195205215225235245255265275285295305315325335345355365375385395405415425435445455465475485495505515525535545555565575585595605615625635645655665675685695705715725735745755765775785795805815825835845855865875885895905915925935945955965975985996006016026036046056066076086096106116126136146156166176186196206216226236246256266276286296306316326336346356366376386396406416426436446456466476486496506516526536546556566576586596606616626636646656666676686696706716726736746756766776786796806816826836846856866876886896906916926936946956966976986997007017027037047057067077087097107117127137147157167177187197207217227237247257267277287297307317327337347357367377387397407417427437447457467477487497507517527537547557567577587597607617627637647657667677687697707717727737747757767777787797807817827837847857867877887897907917927937947957967977987998008018028038048058068078088098108118128138148158168178188198208218228238248258268278288298308318328338348358368378388398408418428438448458468478488498508518528538548558568578588598608618628638648658668678688698708718728738748758768778788798808818828838848858868878888898908918928938948958968978988999009019029039049059069079089099109119129139149159169179189199209219229239249259269279289299309319329339349359369379389399409419429439449459469479489499509519529539549559569579589599609619629639649659669679689699709719729739749759769779789799809819829839849859869879889899909919929939949959969979989991000100110021003100410051006100710081009101010111012101310141015101610171018101910201021102210231024102510261027102810291030103110321033103410351036103710381039104010411042104310441045104610471048104910501051105210531054105510561057105810591060106110621063106410651066106710681069107010711072107310741075107610771078107910801081108210831084108510861087108810891090109110921093109410951096109710981099110011011102110311041105110611071108110911101111111211131114111511161117111811191120112111221123112411251126112711281129113011311132113311341135113611371138113911401141114211431144114511461147114811491150115111521153115411551156115711581159116011611162116311641165116611671168116911701171117211731174117511761177117811791180118111821183118411851186118711881189119011911192119311941195119611971198119912001201120212031204120512061207120812091210121112121213121412151216121712181219122012211222122312241225122612271228122912301231123212331234123512361237123812391240124112421243124412451246124712481249125012511252125312541255125612571258125912601261126212631264126512661267126812691270127112721273127412751276127712781279128012811282128312841285128612871288128912901291129212931294129512961297129812991300



case 'S'

(a) (b) (c) (d) (e) (f) (g) (h) (i) (j) (k) (l) (m) (n) (o) (p) (q) (r) (s) (t) (u) (v) (w) (x) (y) (z) (aa) (ab) (ac) (ad) (ae) (af) (ag) (ah) (ai) (aj) (ak) (al) (am) (an) (ao) (ap) (aq) (ar) (as) (at) (au) (av) (aw) (ax) (ay) (az) (ba) (bb) (bc) (bd) (be) (bf) (bg) (bh) (bi) (bj) (bk) (bl) (bm) (bn) (bo) (bp) (bq) (br) (bs) (bt) (bu) (bv) (bw) (bx) (by) (bz) (ca) (cb) (cc) (cd) (ce) (cf) (cg) (ch) (ci) (cj) (ck) (cl) (cm) (cn) (co) (cp) (cq) (cr) (cs) (ct) (cu) (cv) (cw) (cx) (cy) (cz) (da) (db) (dc) (dd) (de) (df) (dg) (dh) (di) (dj) (dk) (dl) (dm) (dn) (do) (dp) (dq) (dr) (ds) (dt) (du) (dv) (dw) (dx) (dy) (dz) (ea) (eb) (ec) (ed) (ee) (ef) (eg) (eh) (ei) (ej) (ek) (el) (em) (en) (eo) (ep) (eq) (er) (es) (et) (eu) (ev) (ew) (ex) (ey) (ez) (fa) (fb) (fc) (fd) (fe) (ff) (fg) (fh) (fi) (fj) (fk) (fl) (fm) (fn) (fo) (fp) (fq) (fr) (fs) (ft) (fu) (fv) (fw) (fx) (fy) (fz) (ga) (gb) (gc) (gd) (ge) (gf) (gg) (gh) (gi) (gj) (gk) (gl) (gm) (gn) (go) (gp) (gq) (gr) (gs) (gt) (gu) (gv) (gw) (gx) (gy) (gz) (ha) (hb) (hc) (hd) (he) (hf) (hg) (hh) (hi) (hj) (hk) (hl) (hm) (hn) (ho) (hp) (hq) (hr) (hs) (ht) (hu) (hv) (hw) (hx) (hy) (hz) (ia) (ib) (ic) (id) (ie) (if) (ig) (ih) (ii) (ij) (ik) (il) (im) (in) (io) (ip) (iq) (ir) (is) (it) (iu) (iv) (iw) (ix) (iy) (iz) (ja) (jb) (jc) (jd) (je) (jf) (jg) (jh) (ji) (jj) (jk) (jl) (jm) (jn) (jo) (jp) (jq) (jr) (js) (jt) (ju) (jv) (jw) (jx) (jy) (jz) (ka) (kb) (kc) (kd) (ke) (kf) (kg) (kh) (ki) (kj) (kk) (kl) (km) (kn) (ko) (kp) (kq) (kr) (ks) (kt) (ku) (kv) (kw) (kx) (ky) (kz) (la) (lb) (lc) (ld) (le) (lf) (lg) (lh) (li) (lj) (lk) (ll) (lm) (ln) (lo) (lp) (lq) (lr) (ls) (lt) (lu) (lv) (lw) (lx) (ly) (lz) (ma) (mb) (mc) (md) (me) (mf) (mg) (mh) (mi) (mj) (mk) (ml) (mm) (mn) (mo) (mp) (mq) (mr) (ms) (mt) (mu) (mv) (mw) (mx) (my) (mz) (na) (nb) (nc) (nd) (ne) (nf) (ng) (nh) (ni) (nj) (nk) (nl) (nm) (nn) (no) (np) (nq) (nr) (ns) (nt) (nu) (nv) (nw) (nx) (ny) (nz) (oa) (ob) (oc) (od) (oe) (of) (og) (oh) (oi) (oj) (ok) (ol) (om) (on) (oo) (op) (oq) (or) (os) (ot) (ou) (ov) (ow) (ox) (oy) (oz) (pa) (pb) (pc) (pd) (pe) (pf) (pg) (ph) (pi) (pj) (pk) (pl) (pm) (pn) (po) (pp) (pq) (pr) (ps) (pt) (pu) (pv) (pw) (px) (py) (pz) (qa) (qb) (qc) (qd) (qe) (qf) (qg) (qh) (qi) (qj) (qk) (ql) (qm) (qn) (qo) (qp) (qq) (qr) (qs) (qt) (qu) (qv) (qw) (qx) (qy) (qz) (ra) (rb) (rc) (rd) (re) (rf) (rg) (rh) (ri) (rj) (rk) (rl) (rm) (rn) (ro) (rp) (rq) (rr) (rs) (rt) (ru) (rv) (rw) (rx) (ry) (rz) (sa) (sb) (sc) (sd) (se) (sf) (sg) (sh) (si) (sj) (sk) (sl) (sm) (sn) (so) (sp) (sq) (sr) (ss) (st) (su) (sv) (sw) (sx) (sy) (sz) (ta) (tb) (tc) (td) (te) (tf) (tg) (th) (ti) (tj) (tk) (tl) (tm) (tn) (to) (tp) (tq) (tr) (ts) (tt) (tu) (tv) (tw) (tx) (ty) (tz) (ua) (ub) (uc) (ud) (ue) (uf) (ug) (uh) (ui) (uj) (uk) (ul) (um) (un) (uo) (up) (uq) (ur) (us) (ut) (uu) (uv) (uw) (ux) (uy) (uz) (va) (vb) (vc) (vd) (ve) (vf) (vg) (vh) (vi) (vj) (vk) (vl) (vm) (vn) (vo) (vp) (vq) (vr) (vs) (vt) (vu) (vv) (vw) (vx) (vy) (vz) (wa) (wb) (wc) (wd) (we) (wf) (wg) (wh) (wi) (wj) (wk) (wl) (wm) (wn) (wo) (wp) (wq) (wr) (ws) (wt) (wu) (wv) (ww) (wx) (wy) (wz) (xa) (xb) (xc) (xd) (xe) (xf) (xg) (xh) (xi) (xj) (xk) (xl) (xm) (xn) (xo) (xp) (xq) (xr) (xs) (xt) (xu) (xv) (xw) (xx) (xy) (xz) (ya) (yb) (yc) (yd) (ye) (yf) (yg) (yh) (yi) (yj) (yk) (yl) (ym) (yn) (yo) (yp) (yq) (yr) (ys) (yt) (yu) (yv) (yw) (yx) (yy) (yz) (za) (zb) (zc) (zd) (ze) (zf) (zg) (zh) (zi) (zj) (zk) (zl) (zm) (zn) (zo) (zp) (zq) (zr) (zs) (zt) (zu) (zv) (zw) (zx) (zy) (zz)

2500 = 2500

```
SPRA = atoi(argv[4]);
```

```
zFnn = atoi(argv[2]);
```



```
if(G.background) exit(USIBLINDER);
```

setscreen0.curscreen0

```
if(G.main->scene.first==0) {
    sce= add_scene("1");
```

```
set_scene(scene);
```

```
qenter(Q_FIRSTTIME, 1);
```

```
glutMainLoop(); read_file(ar
```

break,

Case 'S':

```
if (++a < argc)
```

Case 8

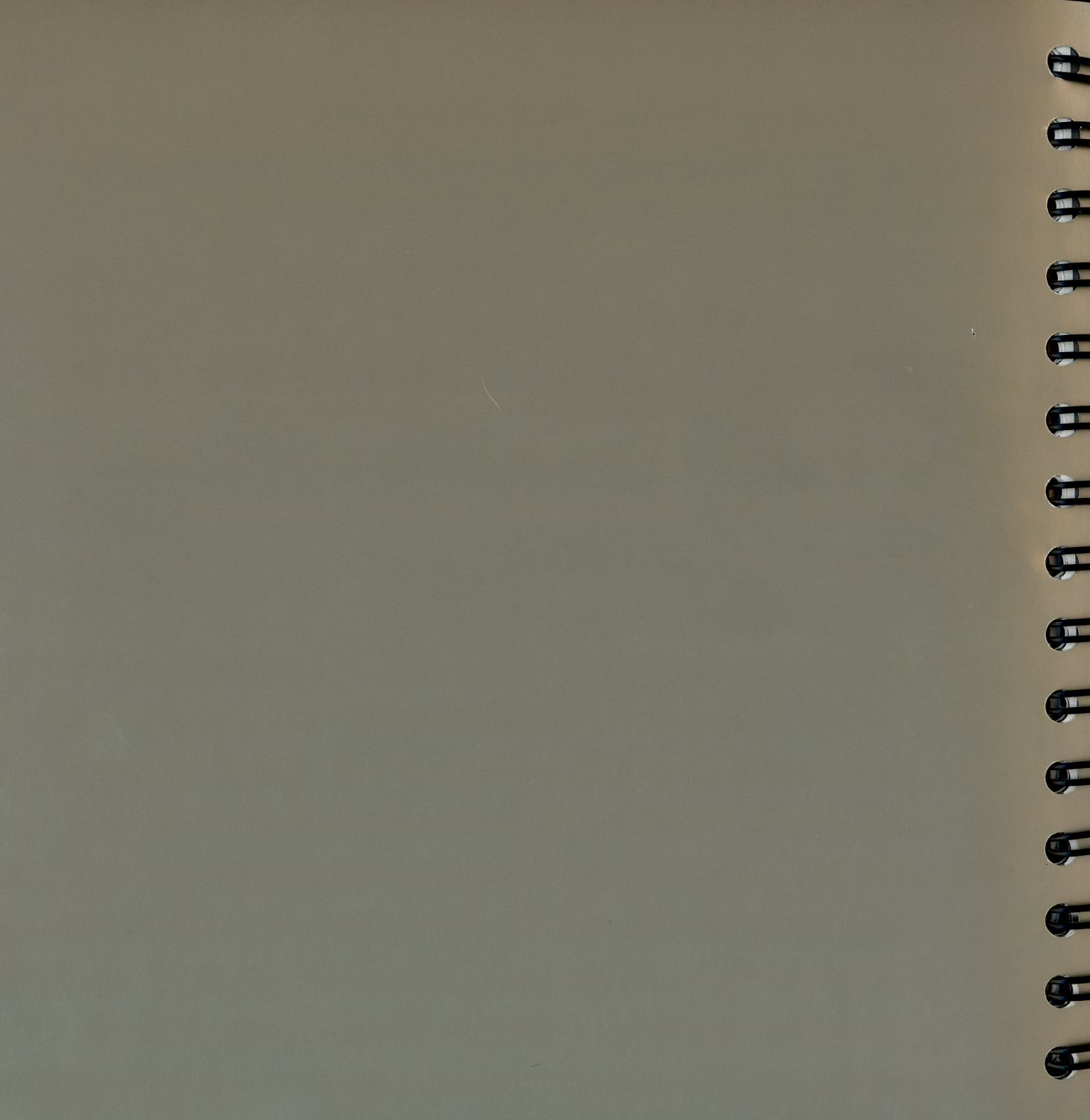
```
setscreen(G curscreen);
```

11/11/11

also of

```
else {
    readfile(argv[1]);
}
```

```
read_file(argv[1]);
```

Installation for Linux or FreeBSD i386 systems.

1 Prerequisites

- Minimal PC configuration:
32MB memory, 16bpp capable video card, 90Mhz Pentium.
- Linux: Slackware 3.4 or Redhat 4.2 / 5.1.
Other versions may work, but are not tested here.
- FreeBSD: version 2.2.6-RELEASE
- XFree86 3.3.x
Again, other versions may work, but have not been tested
- Mesa 3.0
Only the 'dynamic' Blender version needs Mesa installation.
If you don't have Mesa-3.0, get it from:
(Linux rpm) <ftp://ftp.freshmeat.net/pub/rpms/mesa/>
(FreeSBD package:
[ftp://ftp.freebsd.org/pub/FreeBSD/\(source\)](ftp://ftp.freebsd.org/pub/FreeBSD/(source))
<ftp://iris.ssec.wisc.edu/pub/Mesa/>
Due to the ever-changing Internet it is best to check this at our web site.

2a. Download Linux Blender

In the download section of the www.blender.nl site you can get the latest Blender release. There are 2 Linux versions available:

- blenderLinux-dynamic_1.5x.tar.gz
- blenderLinux-static_1.5x.tar.gz

The dynamic version is not including the Mesa library. Only RedHat 4.2 compatible systems can use this version. The static version includes all the libraries you need, and works with Linux 5.1 releases as well.

2b. Download FreeBSD Blender

In the download section of the www.blender.nl site you can get the latest Blender release. There is one version available:

- blenderBSD-dynamic_1.5x.tar.gz

This dynamic version is exclusive Mesa.

3. Uncompress

Use a UNIX shell to move the download file to the directory in which the Blender installation directory should be created.

Make that directory the current directory. Now type:

```
tar zxvf blenderXxxx_1.5x.tar.gz
```

This creates the directory 'blenderXxxx_1.5x'. Initially, it contains the executable and the primary files.

4. Add an environment variable

To be able to find its data files, Blender needs an environment variable that points to its installation directory, or any other directory you wish to copy the installation files to. The executable itself can be copied to any location you wish.

Important: the name_of_directory must be the full path, starting with '/' !

```
setenv BLENDERDIR name_of_directory (for csh shell)
export BLENDERDIR=name_of_directory (for sh shell)
```

You may want to enter this into your .cshrc or .login (or .profile) file, so it is automatically set when you login.

5. Mesa installation

Experiment with these variables for speed. Refer to the Mesa documentation, as well.

- Turn off dithering.
setenv MESA_NO_DITHER
- Explicitely specify your visual
setenv MESA_RGB_VISUAL "TrueColor 24"
- Choose a swapbuffer method, real 24 bit visual
doesn't have proper optimisation in Mesa, in that
case the Ximage method will be extremely slow.
setenv MESA_BACK_BUFFER "Pixmap"
setenv MESA_BACK_BUFFER "Ximage"

6. Start Blender.

Type in the UNIX shell:

```
./blender (if Blender resides in the current  
directory)  
blender (if the PATH environment has been  
correctly set)
```

Within a few seconds Blender is ready to work with. When you run Blender for the first time, it copies the files .B.blend and .B.fs to your home directory. If Blender can't find the installation files, it terminates with an error message. Carefully check whether the environment variable was set correctly.

The object oriented system

Blender has a strictly object-oriented structure. Every aspect of the 3D world has been organized in small data blocks. By linking these together, making copies and editing or re-using them, you can build complex environments with a minimum of memory usage.

The main data blocks

Scene.

This is the 'canvas' of the 3D world. It contains the specific rendering information (camera, image resolution) and the links to Objects. Different Scenes can use the same Objects. Scenes can also be linked together to function as a (film) set.

World.

This block contains sky, stars, exposure, and other environment variables.

Object.

The basic 3D information block. This contains a position, rotation, size and transformation matrices. It can be linked to other Objects for hierarchies or deformation. Objects can be 'empty' (just an axis) or have a link to ObData, the actual 3D information: Mesh, Curve, Lattice, Lamp, etc. Objects can be linked to animation curves (Ipo) and Materials as well.

Mesh.

This is the triangles and quads mesh data. It contains vertices, faces, and normals. It can have a keyframe block for morphing. It can be linked to Materials.

Curve.

Data to be used as Text, Bezier or BSplines and 3D Nurbs Surfaces. It also has a keyframe structure and can be linked to Materials.

Material.

This data block contains visual properties such as colours, reflectivity and transparency. It can be linked to eight different Texture blocks.

Texture.

Use images, procedural formulas or plugins to define textures. Can be linked to Material, Lamp and World blocks.

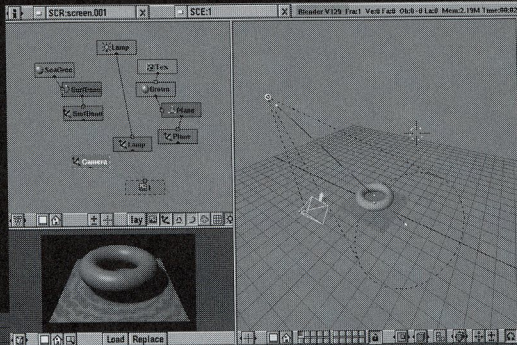
Lamp.

Data to be used for light information, as colours and shadow settings. Can be linked to Texture blocks as well.

Ipo.

This is the main animation curve system. Ipo blocks can be used by Objects (for movement), but also by Materials for animated colours.

Data structure example



This Blender screendump shows a simple Scene. There are three windows in this Screen example:

- The large window is the 3DWindow.
- The square 'diagram window' is called the OutlinerWindow in Blender. It displays the current blocks structure.
- The third window displays the rendered image of this Scene.

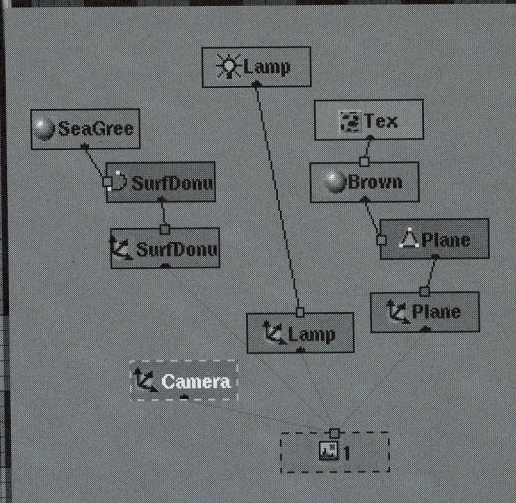
There are four Objects in the 3Dwindow: a Camera, a Mesh plane, a Surface donut and a Lamp spotlight.

All of these Objects have been created, modeled and positioned using the 3DWindow.

Normally, selection in Blender only works on the Objects. Thus, in this example, only four different selections can be made. When you select an Object, the Blender interface immediately visualizes access to the entire structure that has been linked with the Object. Therefore, it's very important to always keep in mind that an Object is only the 'bearer' of a linked structure, a collection of blocks containing information about the actual 3D shape, the rendering properties or animation curves.

This structure is visualized in the OutlinerWindow.

At the bottom is the Scene, named '1'. The Scene has links to the four Objects (the light grey lines).

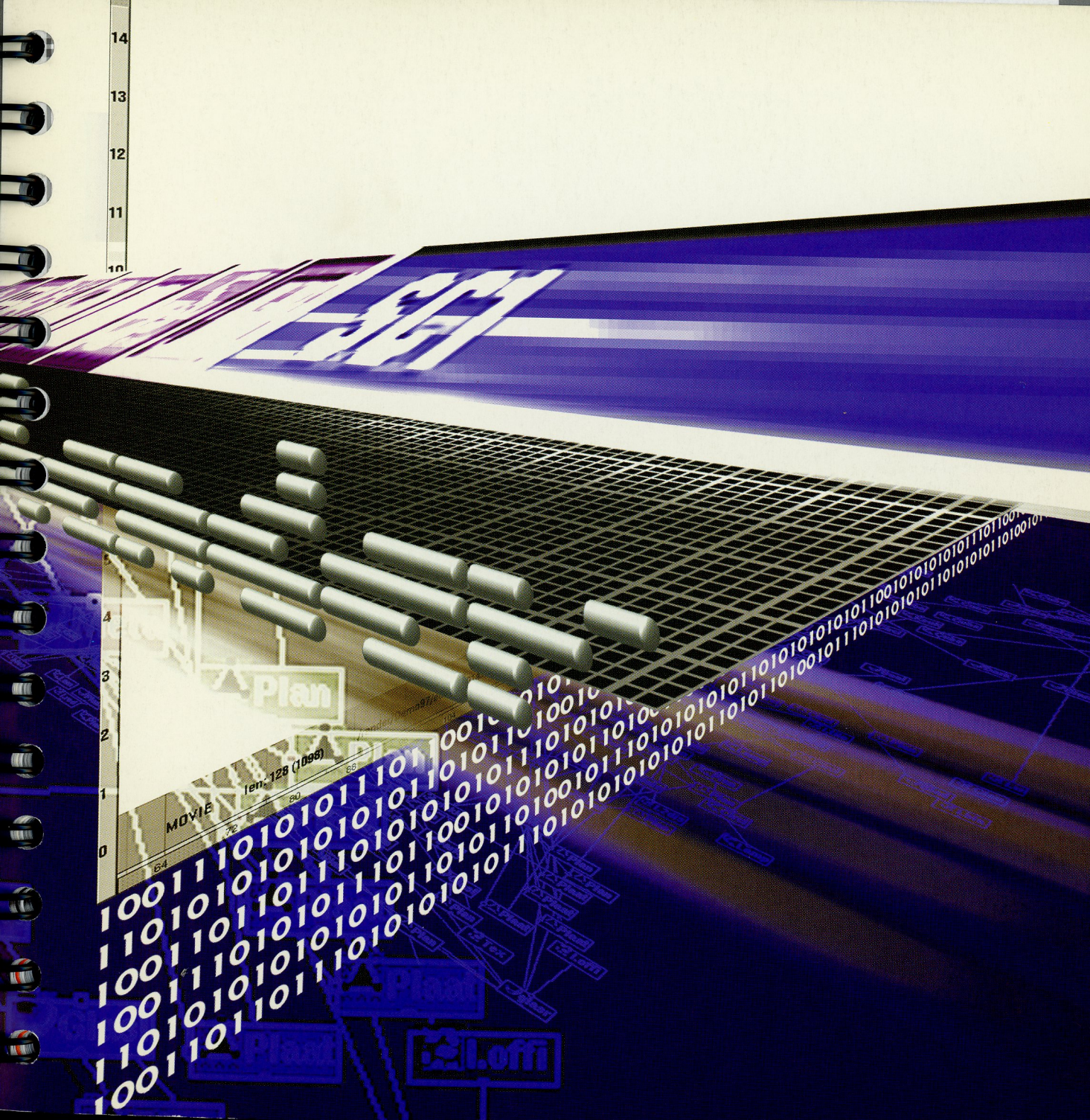


The Object 'Plane' in this example, has a link to the Mesh block 'Plane'. The Mesh block is the actual 3D shape, a single square face that forms the ground for the donut. When you link a Mesh to an Object, the Mesh receives 'existence' and a certain location in 3D space.

The Mesh in turn, has a link to a Material, where the specific rendering properties are stored. Finally, the Material itself is linked to a Texture block.

The Blender interface offers numerous options for creating new blocks, linking these to other blocks and re-using existing blocks.

Getting used to this approach is an important condition for understanding Blender's interface. It is organized in an extremely straightforward and logical system. For many users, this is initially quite confusing and un-intuitive; that's the steep learning curve mentioned before. Luckily, after a while it becomes fun and like 'second nature'; new tools and structures become an adventurous challenge for your creative mind to apply.



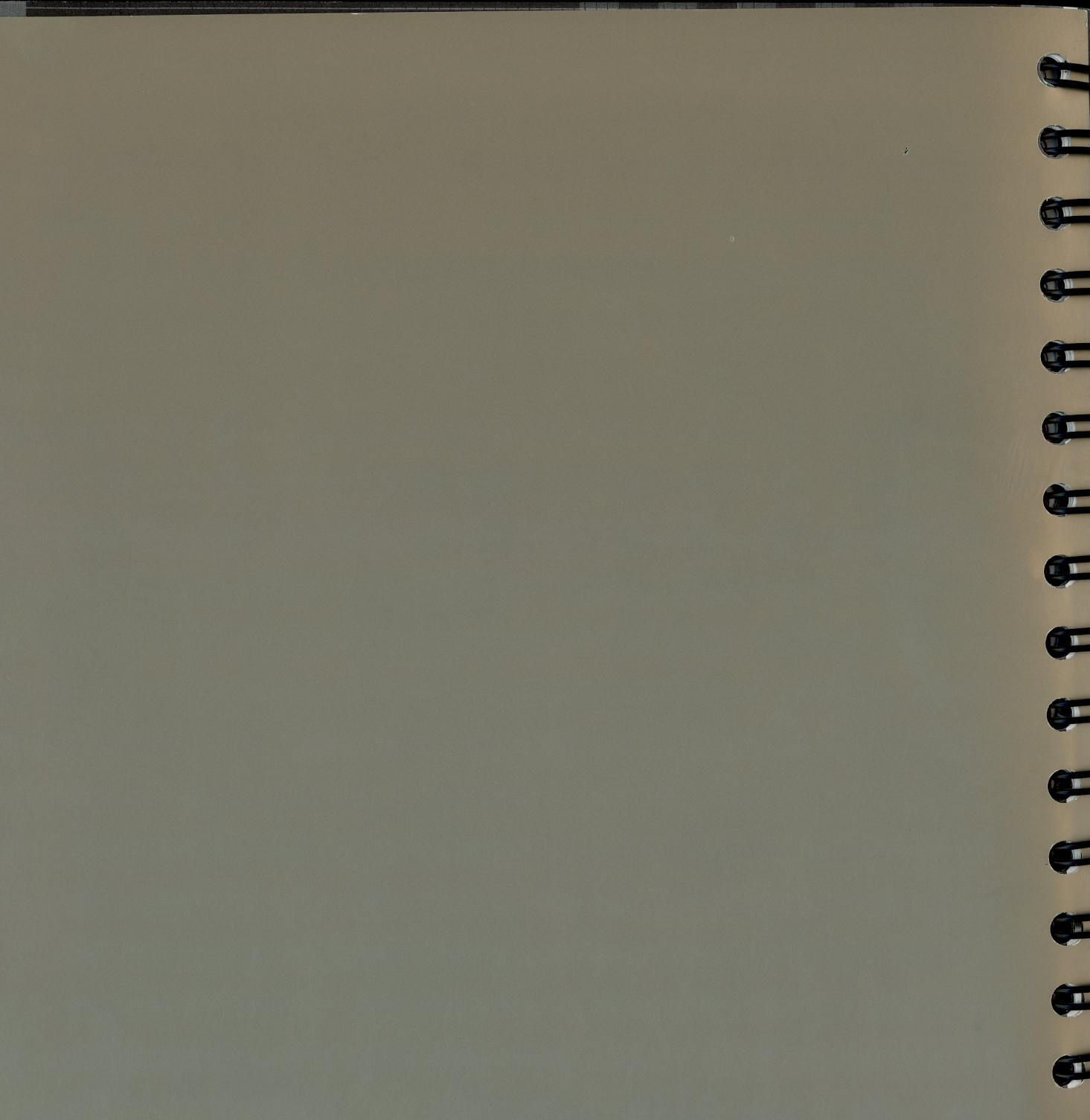
14
13
12
11
10

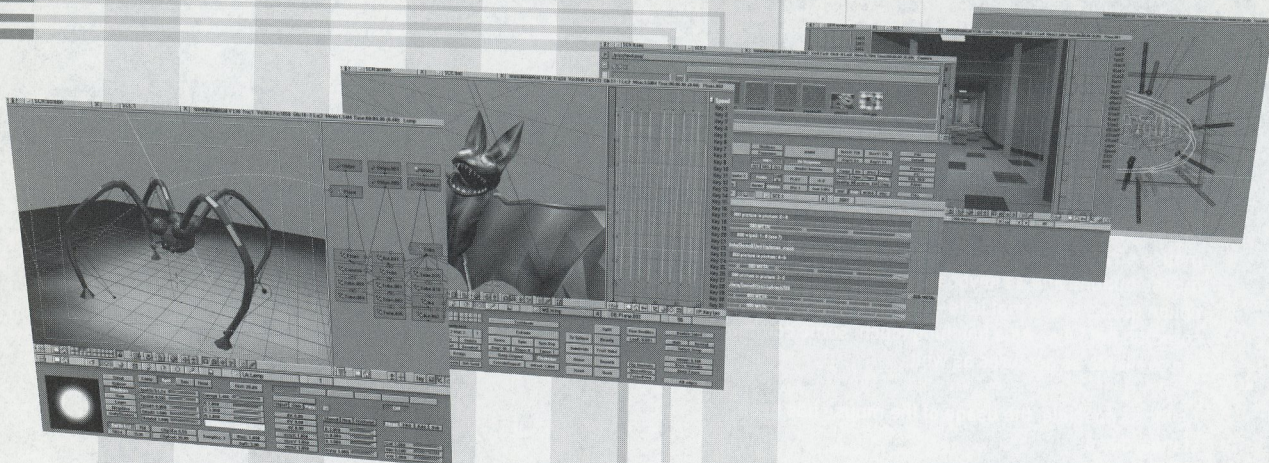
4
3
2
1
0

MOVIE
len: 128 (1088)

Plan

il.offi





Screens and Windows

Each Blender file can have an unlimited number of Screens. A Screen can be subdivided into windows and each window can be given a certain function.

Each window has a header, which can be at the top or bottom at the user's discretion. The leftmost button displays the window type as an icon.

As usual for any window system, only *one* window is active at a time, and only this window can receive the user input. Each window can have a separate mouse and keyboard handling.

To resize a window, simply move the mouse cursor over a window edge. The cursor changes shape. Press and hold LeftMouse to move the edge.

Press MiddleMouse to Split the window.

Press RightMouse to join two adjacent windows.

At the top of a screen usually the Info window is located. It shows the Scene and Screen name and - as text - a lot of statistics. If you pull down the window edge, you find the user options. These will be discussed later.

These are the important window types:

- 3DWindow. It displays Objects as wireframe, solids or Gouraud shaded.
- ButtonsWindow. The visualisation of the specific block types.
- IpoWindow. For animation curves and VertexKeys.
- OopsWindow. The schematic diagram displaying data blocks.
- SequenceWindow. For video editing and post production.
- FileWindow. To load or save files.

Basic editing

The Mouse

Blender is designed as a two-handed system. To minimize mouse strain, almost all commands are available both as mouse buttons and as keyboard hot keys. There's a Toolbox option (press SPACE) that lists – and can activate – the common hot keys. As a finishing touch, some commands are available as gestures as well.

As far as possible, the usage of the mouse has been standardized:

- LeftMouse: for buttons, sliders, text input, placing the 3Dcursor.
- LeftMouse+CTRL: add a vertex.
- MiddleMouse: to change the view or - during transformations - the delimiter.
- MiddleMouse+SHIFT: change view
- MiddleMouse+CTRL: zoom in/out view
- RightMouse: selecting and activating
- RightMouse+SHIFT: extend select.
- RightMouse+CTRL: selecting and activating, only at Object centers.
- RightMouse: hold-move-release: translation mode.

The Keyboard

For the keyboard, these are the common rules:

- F1 - F12: global hot keys, independent of window type.
Here are the most important F-Keys.
F1: Load Blender file.
F2: Save Blender file.
F3: Save image.
F12: Render image.
- Esc: to restore or escape from anything (menu's, editing, rendering, file selecting, etc.)
- Numerical Pad: view related hot keys.
- Spacebar: Toolbox menu, displays a listing of the common hot keys.

All keys are listed in the reference section, part 8.

Selecting

These are the three methods for selection:

- Selecting with **RightMouse** *always* selects (and activates) a single item, it deselects everything else. However, holding **SHIFT** while selecting extends the selection.
- Items can be selected with a 'border' as well: press **BKEY** and draw a rectangle with **LeftMouse** to select, **RightMouse** to deselect.
- Use the hotkey **AKEY** to select all or to deselect all items.

Active and Selected

Blender makes a distinction between *selected* and *active*.

- Only one Object can be *active* at any given time, for example to allow visualisation of data in buttons. The *active and selected* Object is displayed in a lighter colour than other selected Objects.
- Any number of Objects can be *selected* at once. Virtually all key commands have an effect on selected Objects.

This is an important distinction because of the non-blocking interface of Blender. Most hot key actions perform at all *selected* data. But the ButtonsWindow and IpoWindow can only display the contents of *active* data.

Moving / rotating / scaling

Most actions in Blender involve moving, rotating or changing size of certain items. Blender offers a wide range of options in this. See the 3DWindow section for a comprehensive list. The options are summarised below:

Translation

GKEY, Grab mode. Move the mouse to translate the selected items, then press **LeftMouse** OR **ENTER** OR **SPACE** to assign the new location. Press **ESC** to cancel. Translation is always corrected for the view in the 3Dwindow. Use the **MiddleMouse** toggle to limit translation to the X, Y or Z axis. Blender determines which axis to use, based on the already initiated movement.

- **RightMouse**, hold-move. This option allows you to select an Object and *immediately* start Grab mode.

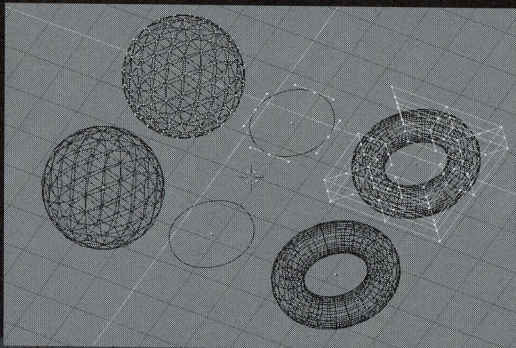
Rotation

- **RKEY**, Rotation mode. Move the mouse around the rotation centre, then press **LeftMouse** OR **ENTER** OR **SPACE** to assign the rotation. Press **ESC** to cancel. Rotation is always perpendicular to the view of the 3DWindow.

Scaling

SKEY, Scaling mode. Move the mouse from the rotation centre outwards, then press **LeftMouse** OR **ENTER** OR **SPACE** to assign the scaling. Use the **MiddleMouse** toggle to limit scaling to the X, Y or Z axis. Blender determines the appropriate axis based on the direction of the movement.

Objects and EditMode



Normally, while working with Objects, you can't change the number of faces or translate some of its vertices. To do this, you must start EditMode by pressing **TAB**. Now editing is locked at the specific *ObData* block of the *active* Object, e.g. the Mesh, Curve or Surface. A new set of tools now becomes available. You can add vertices and faces, extrude polygons, draw curves or assign colours. You can't add or activate other Objects in EditMode. For this you have to leave EditMode by pressing **TAB** again. While entering EditMode, Blender makes a copy of the indicated data. The hotkey **U**KEY here serves as an undo function; actually it restores the copied data. To remind you that you are in EditMode, the cursor shape changes to a cross.

Materials and Textures

In Blender, the Materials and Textures form separate blocks. This approach was selected to keep the interface simple and to allow universal integration between Textures, Lamps and World blocks. You can view the Material settings in the ButtonsWindow, press **F5**. To display the Texture, press **F6**.

The relationship between a Material and a Texture is called the 'mapping'. This relationship is two-sided. First, the information that is passed on to the Texture must be specified. Then the effect of the Texture on the Material is specified.

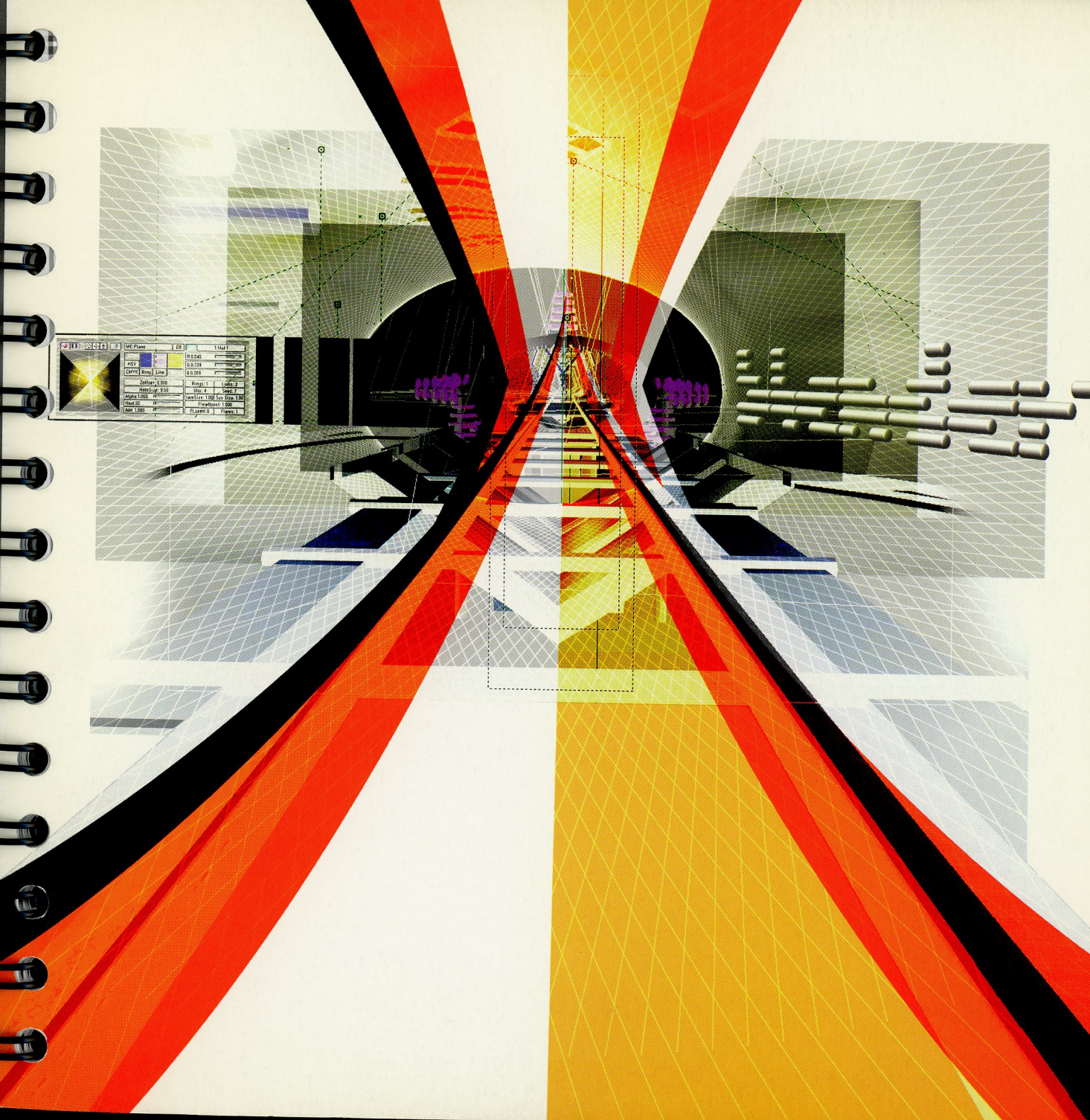
The MaterialButtons on the right-hand side are reserved for the *mapping*.

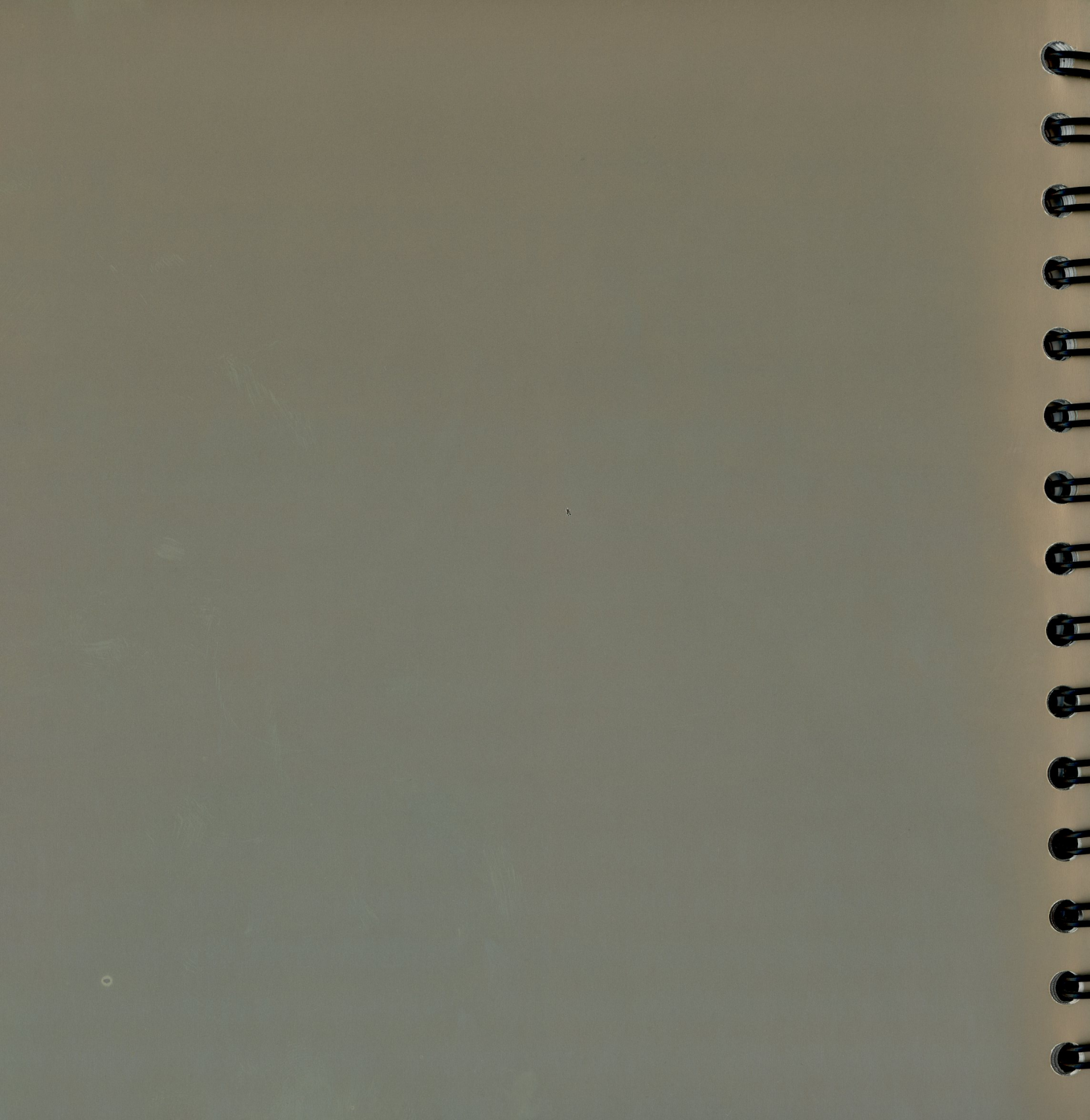
Each Material has eight *channels* to which Textures can be linked. Each *channel* has its own individual *mapping*. By default, textures are executed one after another and then superimposed.

A distinction is also made between 2D textures and 3D (procedural) textures.

"Wood", for example, is a procedural texture. This means that each 3D coordinate can be translated directly into a colour or a value. These types of textures are 'real' 3D, they fit together perfectly at the edges and continue to look like what they are meant to look like even when cut; as if a block of wood has really been cut in two.

The Image texture is the only 2D texture and is the most frequently used and most advanced of Blender's textures. Because pictures are two-dimensional, the way in which the 3D texture coordinate is transformed to 2D must be specified in the *mapping* buttons.





Keys and Animation Curves

Two methods are normally used in animation software to start a 3D object moving.

Key Frames.

Complete positions are saved for certain time units (frames). An animation is created by moving an object through them, interpolated fluidly. The animator works from position to position and can change already created positions.

Motion curves.

Curves can be drawn for each XYZ component for location, rotation and size. These, in fact, form the graphs for the movement, with time set out horizontally and the value set out vertically.

Both systems are completely integrated in Blender: in the "Ipo" system. You can create and edit Ipo's simply by pressing **I**KEY in the 3DWindow. A menu appears with a selection of presets.

To visualize the Ipo block, a special window has to be opened. You can change each window into a IpoWindow with the hotkey **SHIFT+F6**.

The Ipo block in Blender is universal. It makes no difference whether you are controlling an Object's movement or the Material's settings. Once you learn to work with Object Ipos, working with other Ipos is obvious.

Saving and loading files

Use hotkey **F1** to load files, **F2** to save files. The active window then changes into a FileWindow.

- Use **LeftMouse** to activate files or to enter directories.
- Press **MiddleMouse** at a file to load or save it immediately.
- Press **ENTER** to load or save the active file (displayed in the filename button).
- Press **ESC** to cancel the action and restore the previous window type.

The top two buttons in a FileWindow display the directory and the file name respectively. Use **LeftMouse** to activate a button, press **ENTER** to assign the text entry to the button.

There are two ways Blender protects the user for software failures and saving errors:

- File versions: in the UserMenu (top header, pull down the edge) there's a "Versions" button. With this option, before a file will be saved, first the old file(s) will be renamed.

For example: the file `rt.blend` becomes `rt.blend1`.

Auto save: with this option, located in the UserMenu as well, a temporary file will be written every 'x' minutes. The file name has a unique number (the process id). After successfully quitting Blender (press **Q**KEY), this file will be renamed as 'quit.blend'

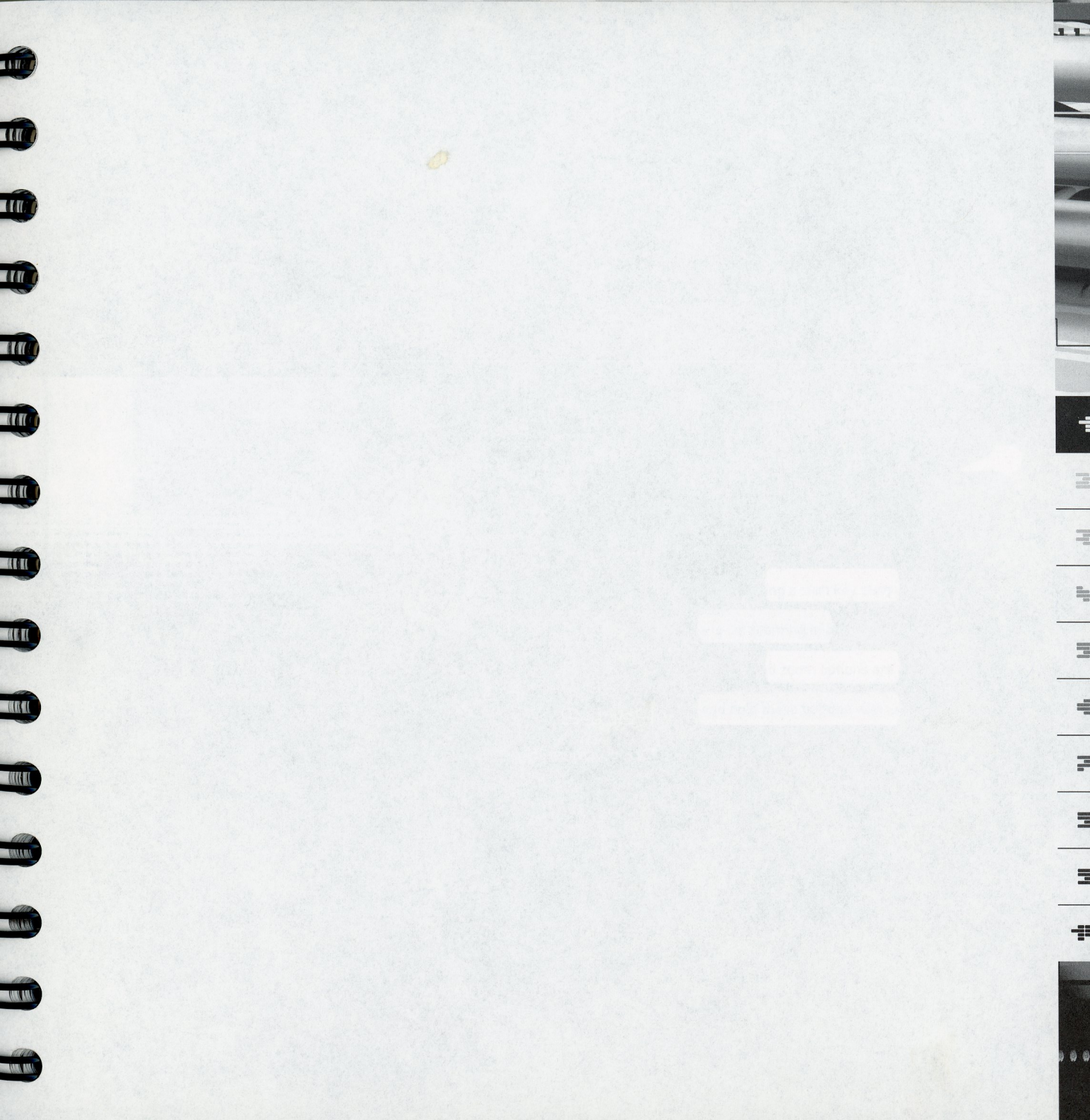
Blender has a - sometimes frustrating - preservative memory management.

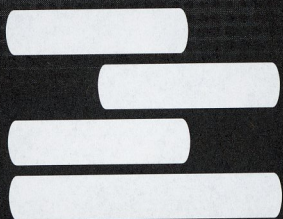
Deleting an Object for example, doesn't mean everything that has been linked (Meshes, Materials...) is immediately released in memory. After a few hours of working, you may have a lot of zero-user data blocks.

This feature can be used effectively to 'undo' a deletion: use the browse button in the headers to restore a link.

Data blocks that have zero users are NOT written in files.

Only when a Blender file is loaded, will memory be released and re-organized.



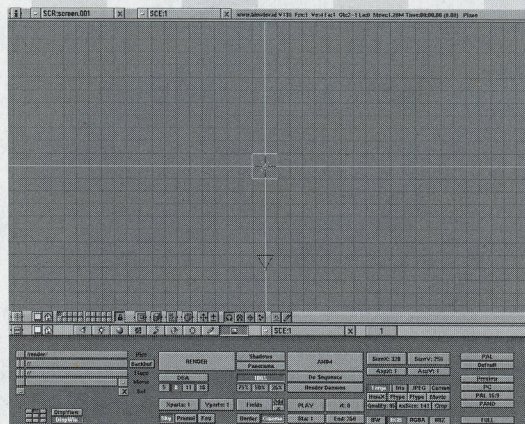


Part 2

Do it yourself

Getting started with 3D

Blender startup



Installation done? Then you're ready to take off. Start up Blender as mentioned in the installation section. Start-up should be finished within a few seconds. What you see is the contents and window layout of the \$HOME/.B.blend file. This file is a regular Blender file and is the default file that loads automatically on start-up. A user can create its own default windows, pre-set Materials, models, directories, etc. You can save the user defaults at any time by pressing CTRL+U.

3D view manipulation

The largest window is the 3DWindow. It displays a Mesh plane (the square), a Camera and a 3Dcursor.

- Click with LeftMouse in the 3DWindow. The 3Dcursor moves to this location. The 3Dcursor is important to indicate where new Objects appear and for rotating and scaling purposes.
- Click, hold and move MiddleMouse in the 3D window. This is the 'trackball' style view rotation. Try this (click-hold-move) at different locations in the window to get used to the TrackBall method.
- Try the same while holding a SHIFT key. SHIFT+MiddleMouse is view translation.
- CTRL+MiddleMouse is for zooming in and out.



The translation and zoom options are also available as buttons in the header of each 3DWindow; press and hold these buttons with LeftMouse, and move the mouse.

All view related commands are located in the numerical pad.

- For rotation: PAD_2, PAD_4, PAD_6, PAD_8 (note the arrows in these keys!).
- For translation, hold SHIFT while using PAD_2, PAD_4, PAD_6, PAD_8.
- PAD_PLUS and PAD_MINUS are for zoom in and zoom out.

If you get lost in 3D space or want to display everything currently visible, press HOME or the header button with the little house.

View pre-sets

The standard pre-sets 'Top', 'Front' and 'Side' view are also available via the numerical pad.

- Top view: PAD_7.
- Front view: PAD_1.
- Side view: PAD_3.

(Note the 'logical' location of the keys in the numerical pad).

The view selected is indicated in the header button.

View modes

The 3DWindow has three modes for viewing:

- Orthonormal, the basic view mode for modeling.
- Perspective. Use the PAD_5 key to toggle Perspective and Orthonormal view mode.
- Camera view: press PAD_0. You can get rid of this view mode by invoking any view command.

Layers



Blender layers are used to organize work and control

visibility. Each Object has a layer setting. Scenes and 3DWindows have layers as well, to determine which Objects are visible.

Currently, there are 20 layers available. Press MKEY to move an Object to a certain layer setting.

Adding Objects

Place the 3D cursor one grid unit above the plane. Do this by switching between Top View (PAD_7) and Front View (PAD_1).

Press SHIFT+A. This calls up the ToolBox in "Add" mode.

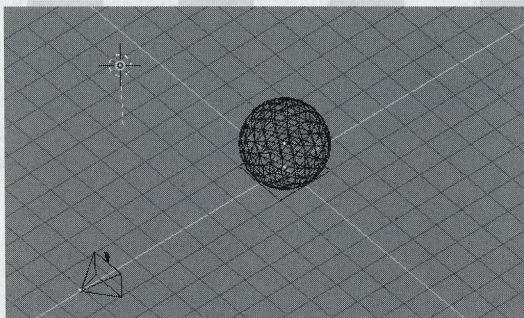
Choose the menu item "ADD→Mesh" by pressing LeftMouse. Now the options change in a list of primitives. Choose "IcoSphere". The ToolBox disappears and a small number requester pops up. Enter a subdivision level by clicking at the left or right side of the button (3 is a nice value). Then press "OK" or ENTER.

Located around the 3Dcursor, a new Object with a Mesh has been added. The Object is automatically in EditMode, so the vertices and faces of its Mesh can be changed. We don't do this now. We leave EditMode by pressing TAB.

You can select these Objects with a RightMouse click, somewhere near the wireframe.

The next step is adding a light. Place the 3D cursor somewhere outside the sphere. Maybe 3-4 grid units away. Press SHIFT+A. The ToolBox always returns as you previously left it. In this case at: "ADD→IcoSphere". To go back to the main "ADD" options, click with RightMouse or move the mouse cursor to the left side of the ToolBox. Then choose:

"ADD→Lamp".



Now a Lamp Object has been added at the 3Dcursor. This type of Object doesn't have an EditMode.

Press `PAD_0` to view from the camera. From this view, Objects can be selected too.

Make a nice composition by using the 'Grabber' press `GKEY`, move the mouse and click `LeftMouse`. The Camera can be moved from this view as well.

When you are satisfied, you can render an image: press `F12`. Rendering a simple scene like this is fast, but you should notice that it is blocking: the mouse cursor changes into a red filled square displaying the current frame number. The rendered image overlaps the 3D view. To get rid of it: press `ESC` or `F11`. To view the image again: press `F11`

It's as easy as that!

Material properties

It is time to do something with the ButtonWindow at the bottom of the Screen.

This ButtonWindow has a header at the top (to shorten mouse movements). In this header you see a row of icon buttons. These buttons indicate the available button groups.



First go to the MaterialButtons. Press the icon button displaying a red sphere.

Then, in the 3DWindow, select with `RightMouse` the Object "Plane". Notice that the Material buttons always shows the active Object.

The complete buttons will be explained in a later chapter. Just for now try to locate these sliders:

- "R", "G" and "B": the colour components.
- "Ref": reflectivity.
- "Spec": specular (shine) component
- "Hard": hardness, the specular size.

The 'Preview' Render in the MaterialButtons immediately displays the result. Push the little red sphere button above it. This indicates a preview option, it doesn't affect the actual rendering.

Back in the 3DWindow, select the Object 'Sphere' (using RightMouse). This newly added Object doesn't have a Material yet. To assign a Material we must use the 'menu' button in the Buttons header:



Press and hold this button. Now a menu pops up with 2 choices:

- 'Material': you can assign the same Material as already used by 'Plane'.
- 'ADD NEW': add a new Material block.

This menu button and its browse functionality is an important Blender feature. You find it throughout the interface for almost all data types: Textures, Meshes, Worlds, Scenes, etc.

Make a quick rendering to look at the new Material appearances (F12). Note that the 'Sphere' still is rendered faceted. Smoothing by normal interpolation is a not a Material property. Actually, it is a Mesh property and can be indicated for each face separately.

Go to the EditButtons (F9) and press the button: "Set Smooth". Make a rendering - F12 - to see the result.

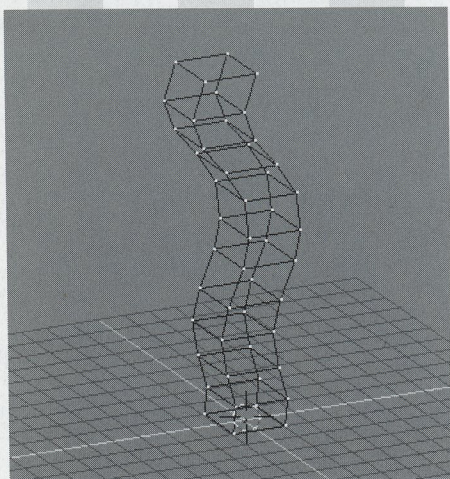
Mesh editing

As mentioned before, the standard editing (such as selecting and translation) can only be done with Objects. To add faces or to move individual vertices, an Object has to be in EditMode.

You enter and leave EditMode by pressing TAB. The cursor changes shape and selection now is possible at individual vertices.

Let's try this. Clear Blender first (CTRL+X). Select the Plane (RightMouse), and press TAB. Blender now is in EditMode.

- Select all four vertices of the plane: use SHIFT+RightMouse,
- press EKEY: a requestor asks "OK? Extrude". Press ENTER.
- Immediately after extrusion, Blender enters 'Grab' mode. Place the newly created vertices at the desired location. Or, if you were working in 'Top View': press ESC, go to Front View (PAD_1), and press GKEY to start the Grabber again.
- You can repeatedly 'Extrude' this shape to create a snake-like form.



Extrusion is a simple and powerful method to create 3D objects. This is how it works:

If the selection contains faces, these are duplicated. At the 'edge' of the selection, new faces are created. Blender automatically deletes 'interior' faces. For example the Mesh created in the previous example is completely hollow.

Blender doesn't have an undo command. To compensate for this, EditMode always saves a copy of the original data. While working in EditMode, the original data can be restored with the command `U`KEY.

In the EditButtons window (press `F9`) you can find a lot of tools that use extrusion, subdivision, or filling. These are described in another section.

Save and Load

To save your work, press `F2`. The current window changes into a FileSelect. The top two large buttons can be used to enter a directory and a filename respectively. You can exit these buttons with `ENTER`.

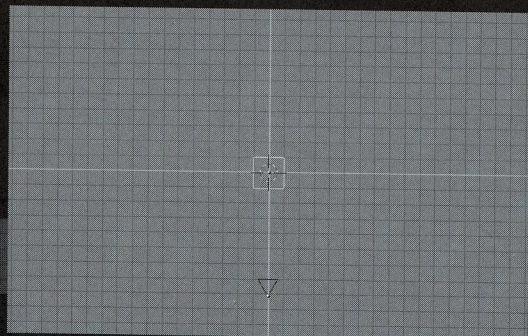
Press `ENTER` again to save the file. If the file exists, a requester pops up to confirm a 'save over'.

A faster way to save without FileSelect is `CTRL+W`.

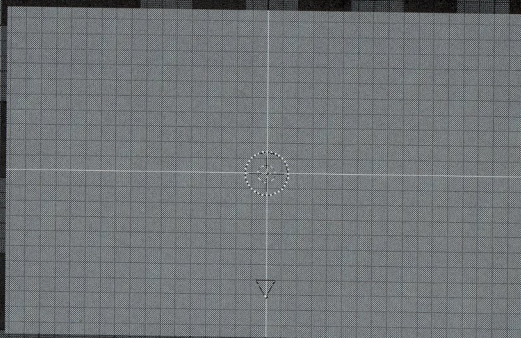
To load a Blender file: press `F1`, move the cursor to a file and press `MiddleMouse` or click `LeftMouse` and press `ENTER`. A faster method to load without FileSelect is `CTRL+O`.

Coffeecup tutorial

This tutorial describes basic Mesh editing. You learn how to model a simple Object, assign it a Material, add light and render an image with shadows.



- The standard Blender Screen displays a Camera and a Mesh plane. The Mesh is pink, indicating that it is active. Pressing **TAB** puts you in EditMode for the Mesh plane.
- First we get rid of the plane, select all vertices with **A**KEY, then press **X**KEY and **ENTER**.
- We add the bottom of the coffeecup by pressing **SHIFT+A**, this calls up the ToolBox. Move the mouse to select the "Circle" option, and press **LeftMouse**.
- The ToolBox disappears and a new requester appears. Press **ENTER** to select "OK".



- Here we see the 32 vertices ('points') which define the circle. Selected vertices are displayed in yellow. The lines that connect the vertices are called 'edges'.
- The circle now will be 'extruded' to become a tube. For this, we first need to change the view: press **PAD_3** for 'Side View'.
- Press **E**KEY and **ENTER** to invoke "Extrude". Blender automatically starts translation mode. Move the mouse to place the newly created circle 5 units up. Press **LeftMouse** to stop translation mode.



O ____

→ 8000~

O ____
H ____

→ 7000~

Na ____

→ 6000~



Fe ____
Mg ____

→ 5500~

→ 5000~

→ 4800~

H ____

→ 4600~

Fe ____

→ 4400~

→ 4200~

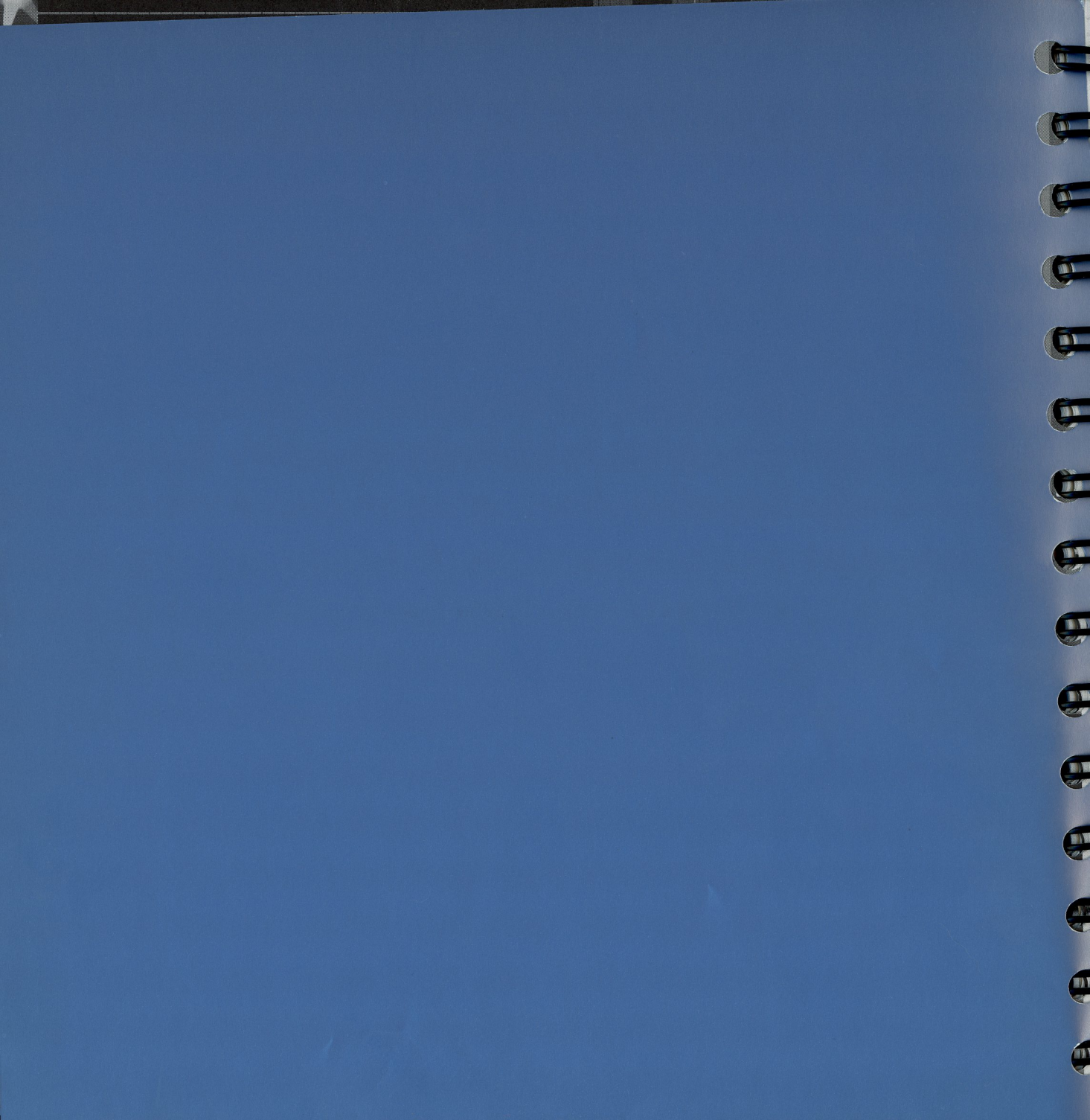
Ca ____

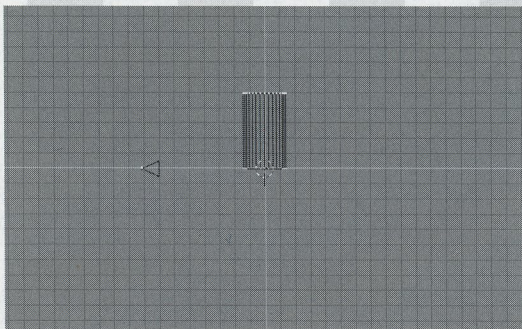
→ 4000~

Ca ____

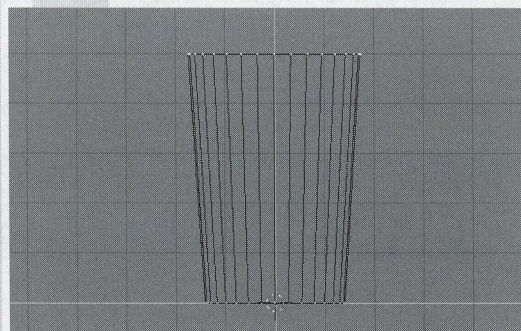
→ 3800~



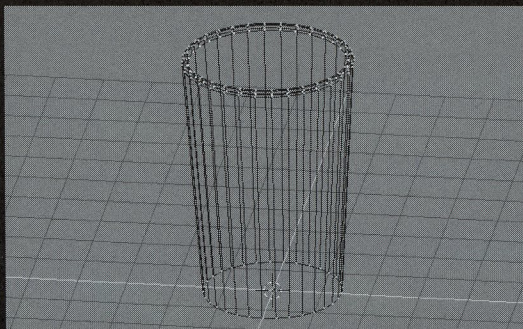




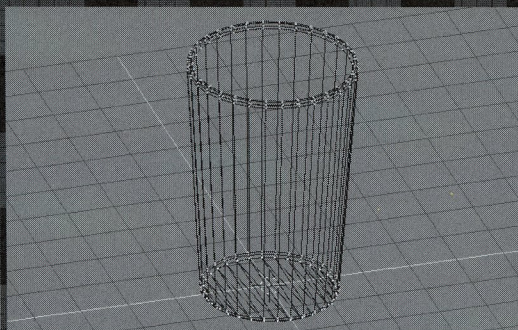
- We now have created a hollow tube. The "Extrude" command creates faces for all selected edges. You can use MiddleMouse (hold-move) to view this. Return to 'Side View' by pressing PAD_3 again.
- Use the scaling mode, SKEY, to slightly enlarge the top of the tube. Move the mouse and press LeftMouse to stop scaling mode.
- To create the soft rounded edge for the cup, we had better zoom in at the tube. Use SHIFT+MiddleMouse to translate the view, and use CTRL+MiddleMouse to zoom in.



- Now we add 4 small extrudes to make the rounded edge.
Press EKEY, ENTER and move the newly created vertices a few pixels up with UPARROWKEY, then press SPACE to end translation mode.
- Repeat the previous step again. Then press SKEY and scale the vertices a few pixels smaller, again by using one of the ARROWKEYS. Press SPACE to end scaling mode.
- Press EKEY, ENTER again, press SPACE to end translation mode. Press SKEY and scale the vertices a few pixels smaller, again by using one of the ARROWKEYS. Press SPACE to end scaling mode.
- The fourth extrude is like the first step, now we translate the new vertices a bit lower and scale the vertices smaller.

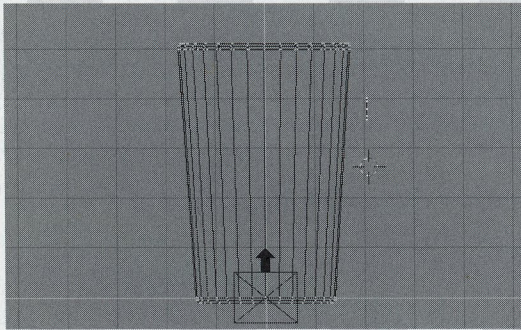


- When you rotate the view with **MiddleMouse**, it should look something like this! Return to Side View (**PAD_3**)
- To create the bottom, we deselect all vertices (**AKEY**) and use **Border Select** to select the lower part of the tube (draw a rectangle using **LeftMouse**-hold). Then add two small extrudes to make a rounded edge.
- After the second extrude, with the vertices still selected, press **SHIFT+F** to fill the bottom. Rotate the view with **MiddleMouse** to see what happened.

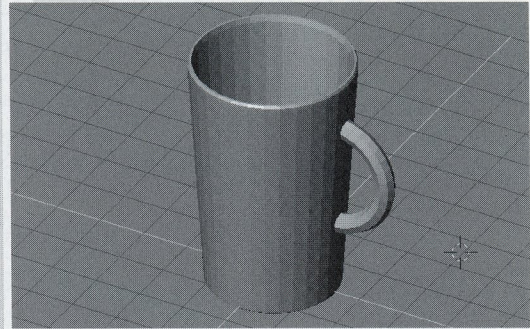


- The next step is creating the handle. We will use the "Spin" tool for this.
- Go to Side View, **PAD_3**. Locate the 3Dcursor to the right of the tube (**LeftMouse**).
- Press **SHIFT+A** and choose "Circle" again. Now we use the requester to fill in a smaller value. With **LeftMouse**, press and hold the button and move the mouse slightly to the left. This is the fast way to assign values to 'Number Buttons'. For this circle, 8 vertices is OK.
- Scale the circle smaller, about 15% of its original size. (**SKEY**).
- Now we go to Front View, **PAD_1**. You can see the circle in the middle of the tube. Move it to the right with the 'grabber'. **GKEY**.

Place the 3Dcursor one grid unit below the circle
(LeftMouse).



- The Spin command is located in the EditButtons (F9). Press the salmon colored "Spin" button twice. In the 3DWindow, you can see a repetitive extrude of the circle, rotating around the 3Dcursor.
- Select all the vertices of the handle (LKEY with mouse cursor near the vertices) and move the handle to the correct location (GKEY). You also need Top View (PAD_1) to do this.



- Leave EditMode (TAB) and press ZKEY to view a solid display of the Object. Press ZKEY again to restore wireframe drawing.
- The cup is quite big. scale it down half its size (SKEY).
- Press the button EditButtons → "Set Smooth". When you use this option *in* EditMode, only the selected faces get 'smoothing' now the whole Mesh is set to 'smooth'.
- Press F5 for the MaterialButtons. There is already a Material available. Mix a nice green colour for the cup to lower the values of the "R" and "B" sliders.
- Now you can add a light, press SHIFT+A. Notice that the ToolBox always returns at the last item choosen. To find the "ADD Lamp" item, press RightMouse once or move the mouse cursor to the left side of the ToolBox. The lamp is added at the 3Dcursor location. Zoom out (PAD_MINUS) and use GKEY to move the lamp to a location outside and to the left-front side of the cup.
- Press F12 to see the rendering. Press F11 to push the render window to the back.

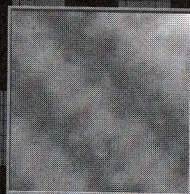


- We are not finished yet. First we add a plane to serve as a table for the cup. Go to Top View (PAD_7) and press SHIFT+A, "ADD→Mesh→Plane". Leave EditMode (TAB) and use the grabber (GKEY) to place the plane at the correct location. For this you need two views, PAD_7 and PAD_1 for example.
- Scale the plane a lot larger using SKEY.

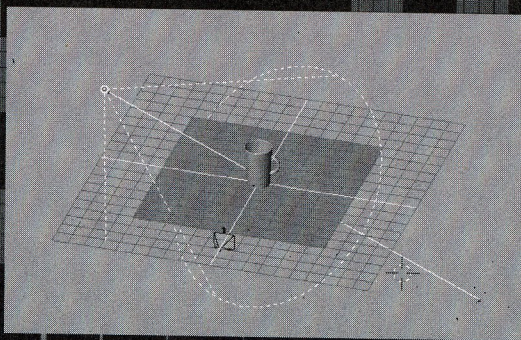


- The MaterialButtons (F5) don't show a Material now. Use the small square 'Menu button' in the header to add a new Material, press and hold the button and choose "ADD NEW". Assign the Material a brownish colour.
- To add a texture to the table, press F6 for the TextureButtons. Here you see the same 'Menu button' again. Use this (choose "ADD NEW") to add a new Texture block. This block then is automatically assigned to the Material of the plane.

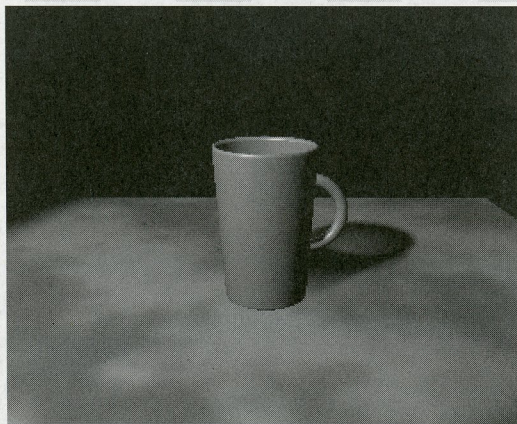
- Press the grey button "Marble" to set the type of texture.
- Press F5 for the Material Buttons again. We now see an ugly purple colour mixing with the brown Material colour. You can change the purple colour with the RGB sliders on the right hand side of the menu. Make it a lighter shade of brown.



- The Lamp is still a point source. To enable shadows for light, we have to change the Lamp to a spotlight.
- Select the Lamp using RightMouse and press F4 to display the LampButtons. Press the green button "Spot". Now the Lamp is displayed differently in the 3DWindow. Use the Front View (PAD_1) and Side View (PAD_3) to rotate the lamp's bundle to point it at the cup. For nice shadows, you can also move the Lamp to a higher position (GKEY).



- Change the spot bundle to make it softer, use the LampButtons→SpotBl slider for this.
- Because we use only a single light at this rendering, increase the Lamp's "Energy" value (set it at 2.0).
- Go to Side View (PAD_3) and move the camera up a bit (GKEY), and rotate it (RKEY) to view the Scene in its entirety. Use PAD_0 to check what the Camera sees.
- Press F10 for the DisplayButtons and select the "Shadows" button in the center to activate shadow rendering. Now hit F12 to view the final rendering.

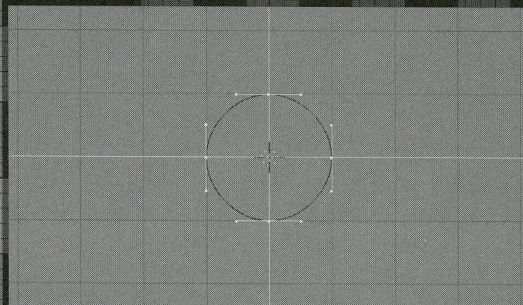


Flying logo tutorial



This tutorial describes basic Curve and Font editing. We also make a start with a simple animation.

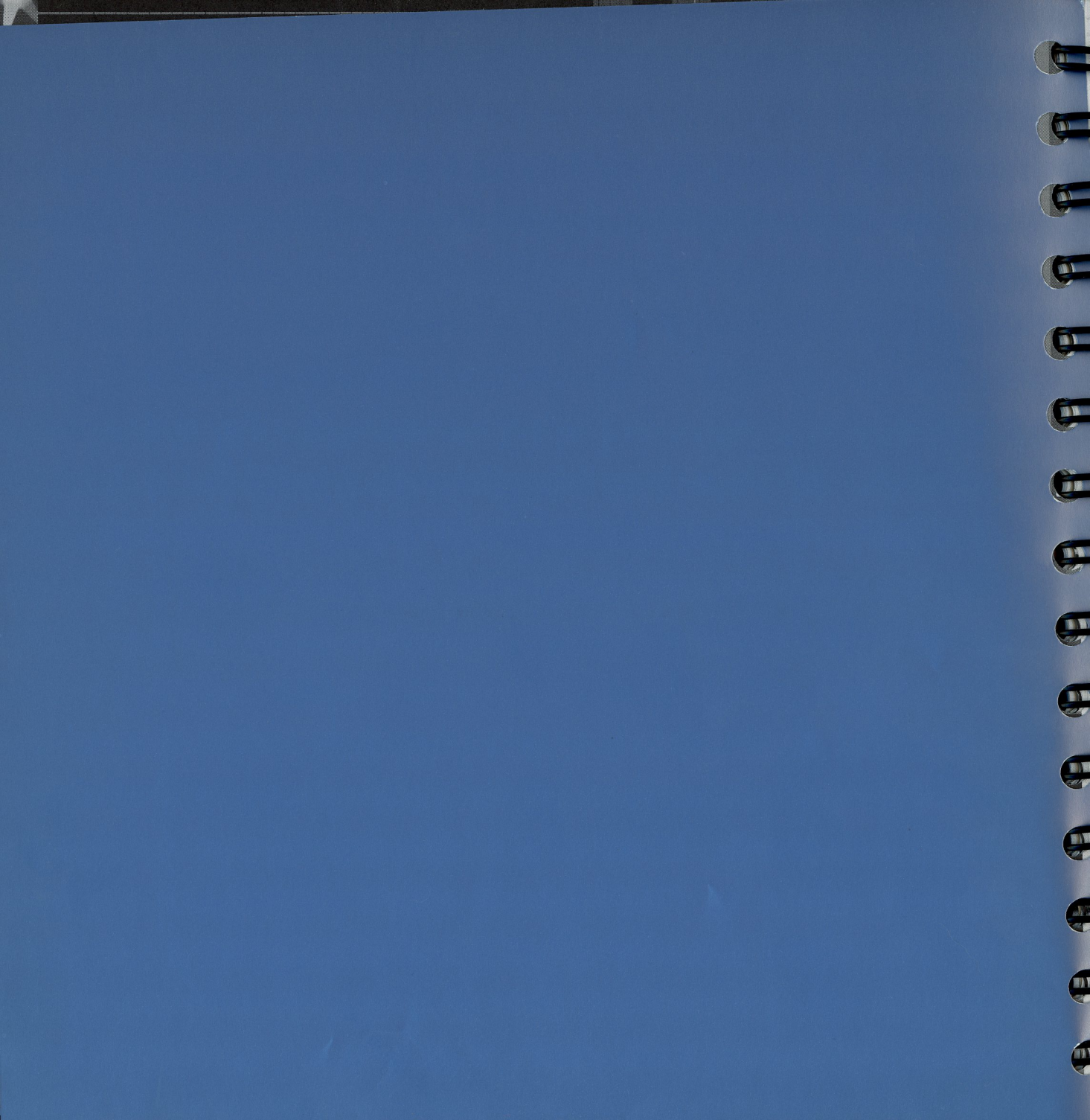
- Start with a clean Blender. You can do that by pressing **CTRL+X**. Now all memory is released and the startup file '.B.blend' is reloaded.
- Delete the standard Plane, press **XKEY**, **ENTER**.
- Go to Front View (**PAD_1**). The Camera is displayed in the center of the 3DWindow.
- To have a clean 'work space', we activate layer '2' by pressing **2KEY**.
- Flat (text) logos are best modeled with Curve Objects, using Bezier curves. Press **SHIFT+A** and select "**ADD→Curve→BezierCircle**".

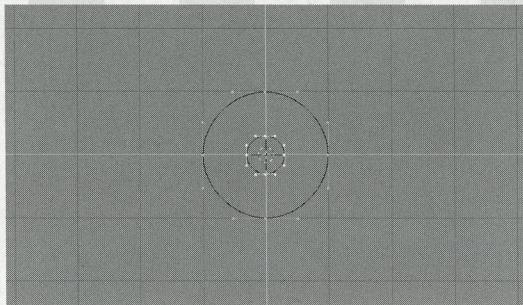


- Bezier curves are defined by 'handles'. Each handle consist of three vertices. By selecting the handle or one of its vertices it is easy to define the curvature. We draw and edit curves later, for this logo we start with the 'O'.
- With all of the vertices of the circle selected we make a duplicate: press **SHIFT+D**. Blender automatically starts 'grab mode', you can switch this immediately to 'scale mode' by pressing **SKEY**.

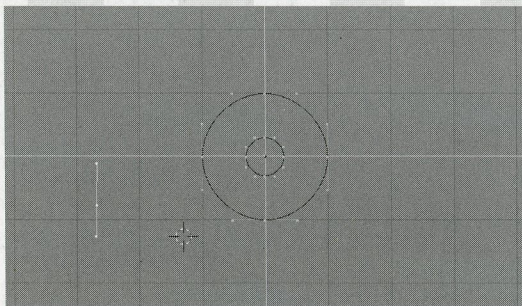
Scale the new circle 30% of the original size. Press **LeftMouse** to end scaling mode.



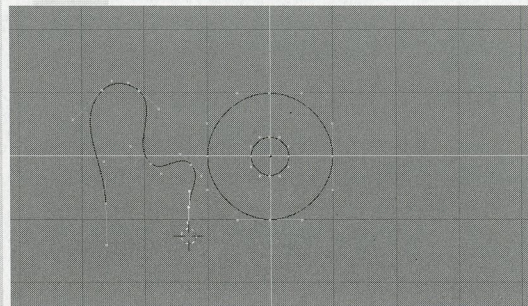




- Leave EditMode (TAB) and press ZKEY to display the result. Curves, including holes, are automatically filled.
- Press ZKEY and TAB again.
- Next we draw a rounded 'L'. You can start a new curve by selecting a single handle of the circle with RightMouse, and press SHIFT+D. Move the mouse to the left and position the newly created handle as indicated:



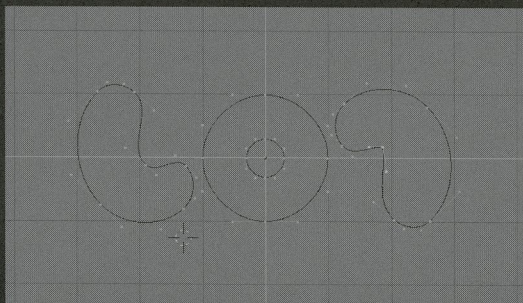
- Now, add new handles to this curve with CTRL+LeftMouse clicks. Do this 5 times to make the rounded 'L' shape.



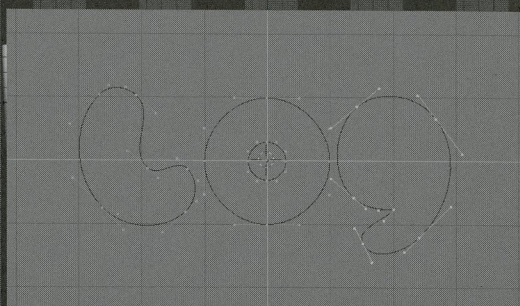
- The yellow colour of a handle indicates this handle has 'auto type'. Select a handle at the center vertex and press GKEY. If you move the mouse, you can see the curvature being automatically recalculated. Press ESC to restore the old position.
- You can easily reposition handles to the desired positions using this method. Press CKEY to make the curve cyclic. Only cyclic curves can be filled.

We make the letter 'G' with a copy of the 'L'. Select the letter with the mouse cursor near the curve using LKEY ("select Linked"). Press SHIFT+D to make a duplicate, locate it to the right side of the 'O'. Press RKEY and rotate the curve 180 degrees while holding CTRL.

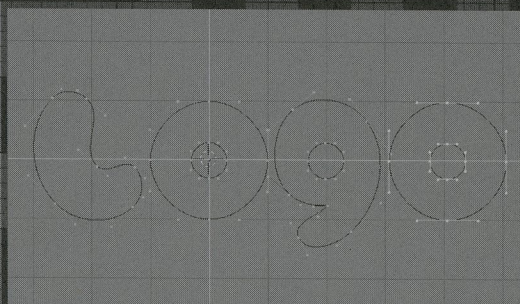
- The 'tail' of the 'G' needs a sharp angle, select this handle at the center vertex. Then press VKEY to change the handle type to 'vector'



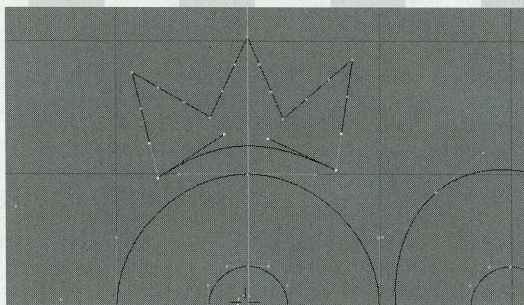
- The handle now is 'broken', you can manually adjust the handle by selecting one of its endpoints and moving it to a new position (GKEY).
- Note that the 'vector' handle then automatically degrades to another type: the black coloured 'free' handle.
- The same is true for the yellow 'auto' handles. When you move one of its endpoints, the type degrades to a pink coloured 'aligned' handle.
- The hotkeys for handle types are:
 - SHIFT+H: auto handle
 - HKEY: free or aligned handle (toggle)
 - VKEY: vector handle.
- The easiest method for moving handles is with the combined selecting-grab gesture: select a vertex with RightMouse, keep RightMouse pressed and move a few pixels, then release RightMouse after 'grab mode' starts automatically.
- Now edit the handles to create the desired 'G' shape.



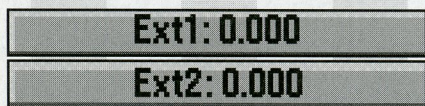
- Press AKEY to deselect all. Select with LKEY (mouse cursor near the curve) the center circle of the 'O'. Make a duplicate (SHIFT+D) and move the new circle to the center of the 'G'. You may want to adjust the outer 'G' line to smoothly surround the center circle.
- The second letter 'O' can be duplicated in the same way. Now duplicate both the circles of the first 'O'.



- To create the last element, the 'crown', we duplicate one of the handles and place it somewhere diagonal above the first 'O'.
- Press VKEY to make the handle 'vector'.
- Then using a series of CTRL+LeftMouse clicks you can draw the 'crown'. Press CKEY to make it cyclic.
- The bottom line of the 'crown' needs a curve to match the bend of the 'O'. Select the endpoint of the handles and move it to create the bend. Note that a single handle can have two types, one for each half.



- Leave EditMode (TAB) and open the EditButtons (F9).
- Locate the grey buttons "Ext1" and "Ext2". These are 'number buttons' and display the extrusion and beveling width respectively.



- You can type in a value in 'number buttons' by pressing LeftMouse+MiddleMouse at the button. The button then changes to a 'text button'. Press SHIFT+BACKSPACE to clear the contents. Type in '0.2' and press ENTER to assign the value. The button "Ext2" can be assigned '0.06' using the same method.
- Press ZKEY to display the Curve Object as a solid.
- Press PAD_0 for Camera View. The Curve is quite big and doesn't fit in the frame. Scale it down (SKEY) and translate it to the center (GKEY).

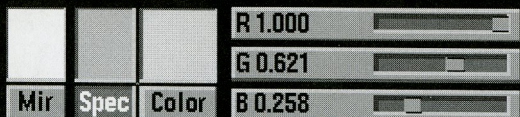


Now we will make two Materials for the Curve Object. Press F5 to view the MaterialButtons. There is no Material available yet.

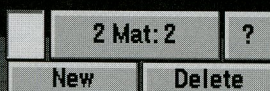


Press the small square 'menu button' in the header and choose 'Material'. The 'zero' in front of the name indicated there were no 'users' for this Material, in fact it was the Material used by the Plane we removed previously.

- Press the little 'car' icon in the header to assign an automatic name to the Material ("Grey"). For a 'metallic' look we assign this Material a coloured specular. Press the small "Spec" button to make the RGB sliders displaying the specular colour. Lower the 'G' and 'B' sliders to mix an orange colour.



- Objects in Blender can have up to 16 different materials. To assign the 'crown' a different colour we add a second Material 'index' with the button EditButtons→New (F9). The button above the "New" button now indicates there are two Material indices, with the second index active.

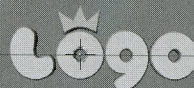


- To decrease or increase the active index you can press to the right or left hand side of this button. For now, we want index '2' to be active.
- Enter EditMode (TAB) and select the 'crown' curve with LKEY.
- Press EditButtons→Assign to assign 'index number 2' to this curve. Leave EditMode again (TAB).
- When you return to the MaterialButtons (F5) you can see this situation displayed in the header:



- The buttons have changed to a blue-grey colour to indicate this Material has more users. The small button '2' indicates the amount. In fact, by pressing EditButtons→New, we have created a Curve Object with two links to the same Material "Grey"
- Press the "x" button to make this link "Single User". Now a full duplicate is made of the Material, named "Grey.001".
- Assign this new Material a yellow-orange colour.
- Remember to press the "Color" button to make the RGB sliders displaying the basic reflective colour.
- Press the 'car' icon button to assign the Material a name.

- Press ZKEY once or twice to view the results. The crown now is displayed in the colour of the second Material.



- The next step is adding two lights.
- Press SHIFT+A, press RightMouse to return to the main menu and choose "ADD→Lamp".
- Go to Side View (PAD_3) and move the Lamp to the front of the Curve Object (GKEY).
- Press SHIFT+A, ENTER to add a second lamp. Locate this Lamp somewhere aside. This Lamp will be used to boost the beveling gloss.
- Press F4 for LampButtons, this now displays the settings for the second Lamp. Assign it a light blue colour. Then press the green button "Sun".
- With the mouse cursor in the 3DWindow, press ALT+R to clear the Lamp's rotation. It now points straight down, thus only illuminating the bevels.
- Press F12 to view a rendering.

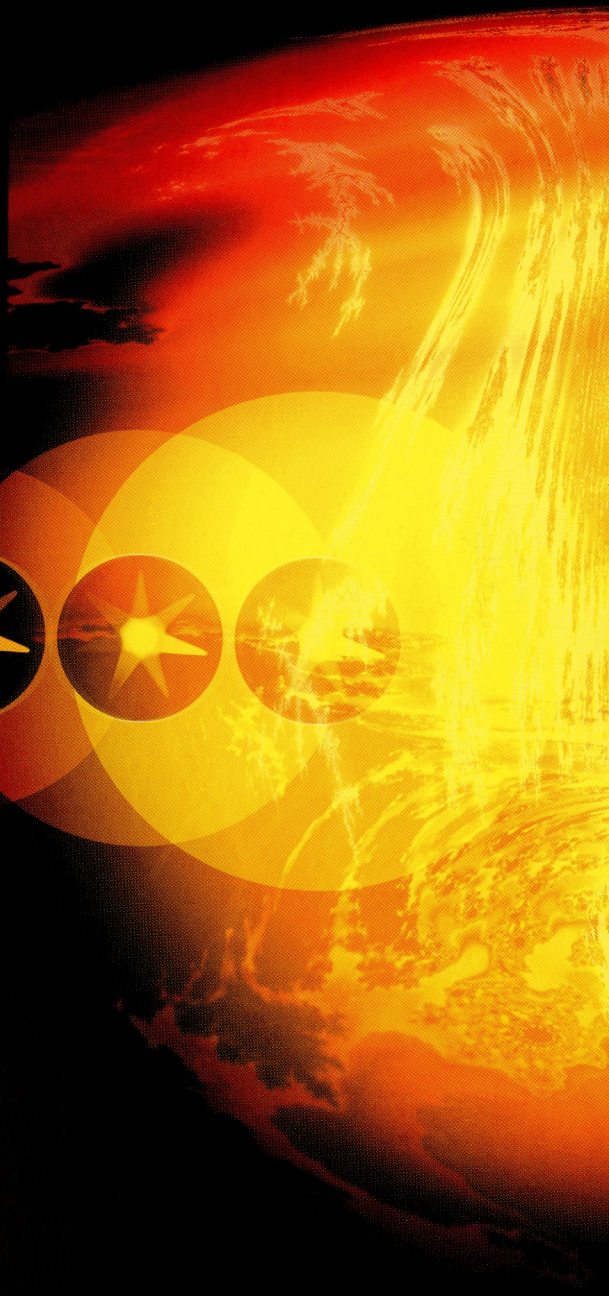
01-02-03-04-05-06-07-08-09-10	11-12-13-14-15-16-17-18-19-20
21-22-23-24-25-26-27-28-29-30	31-32-33-34-35-36-37-38-39-40
41-42-43-44-45-46-47-48-49-50	51-52-53-54-55-56-57-58-59-60
61-62-63-64-65-66-67-68-69-70	71-72-73-74-75-76-77-78-79-80
81-82-83-84-85-86-87-88-89-90	91-92-93-94-95-96-97-98-99-100

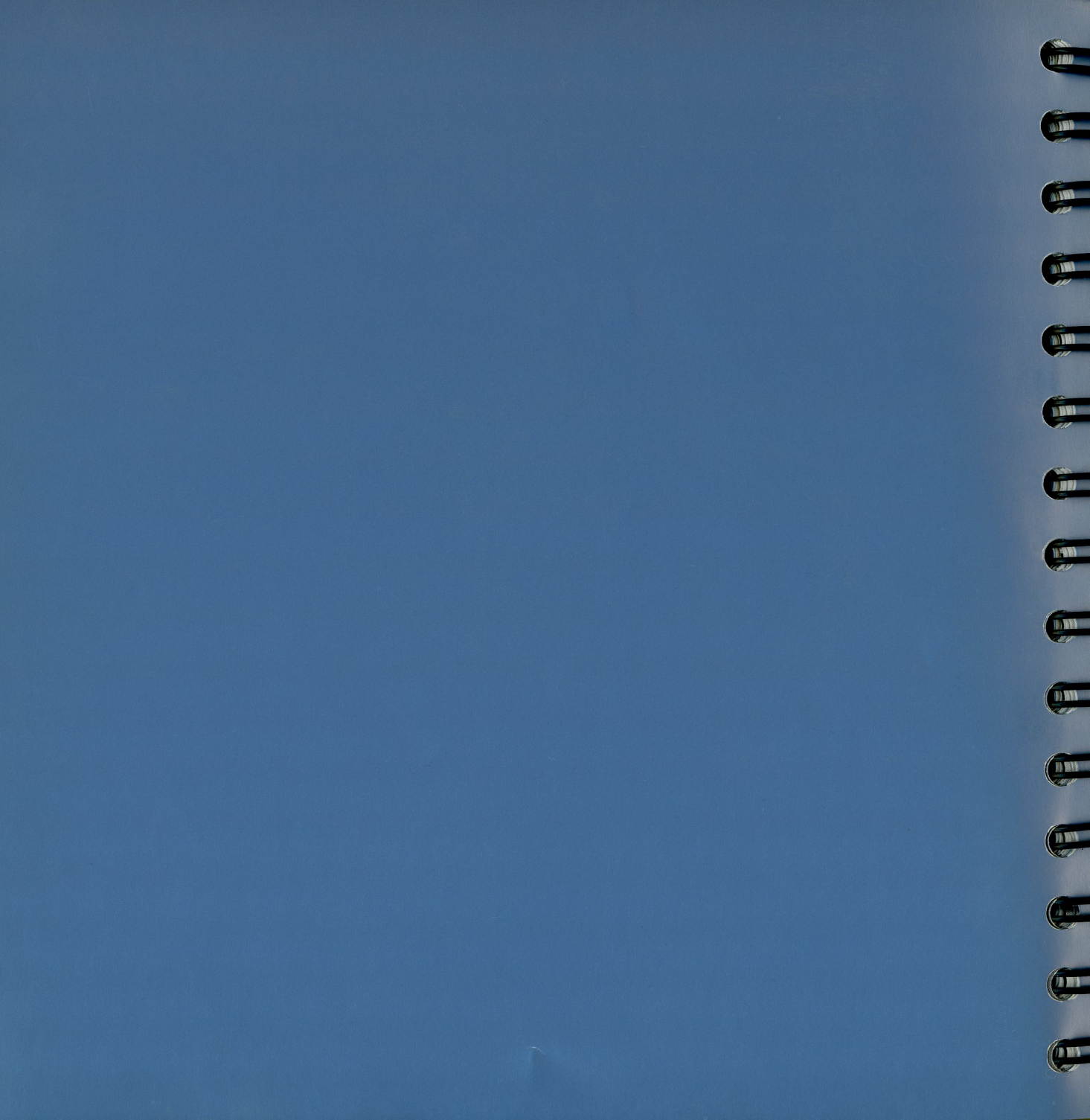
STAR **SUN**

SPECTRAL TYPE **G2**

SURFACE TEMPERATURE **6000⁰**

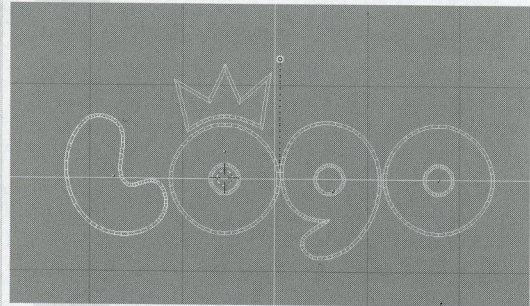
SPECTRAL CLASS	1	O - STAR	/	35 000 ⁰
	2	B - STAR	/	22 000 ⁰
	3	A - STAR	/	14.000 ⁰
	4	F - STAR	/	10.000 ⁰
	5	G - STAR	/	6 000 ⁰
	6	K - STAR	/	4 500 ⁰
	7	M - STAR	/	2 500 ⁰







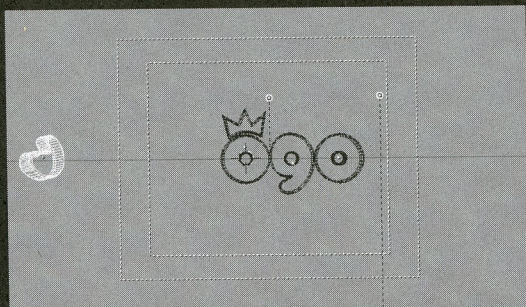
- The animation we want to make is fairly simple, we want each of the letters to move in from both sides.
- We can't do this now, first we have to separate the Curve into four pieces.
- Press ZKEY to restore wireframe drawing.
- RightMouse-select the Curve Object and enter EditMode (TAB).
- Using LKEY, select the 'L' curve. Then press PKEY ("separate") to place the curve into a new Object.
- Do the same with both curves of the 'G' and both curves of the second 'O'.
- Leave EditMode (TAB). Now you can select each Object individually.
- Note that the Object centers are still at the original location.
- We can relocate an Object center as follows. Select all Curve Objects with BKEY; draw a rectangle with LeftMouse. Then press (F9) EditButtons → CentreNew.



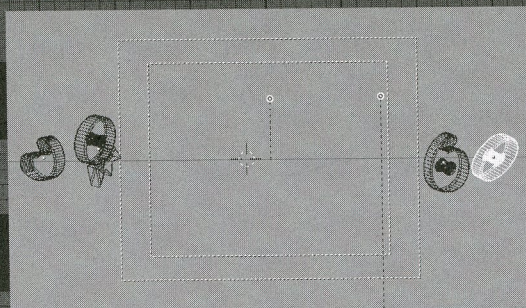
- We want the animation to have 50 frames. Press F10 for the DisplayButtons and change the "End" value to 50.
- Move the 'current frame' to 50 by pressing SHIFT+RIGHTARROW.
Now we are ready to insert the first 'key position' for the logo.
- Press PAD_0 for Camera View.
- Move the mouse cursor to the 3DWindow, and still with all four Curve Objects selected, press IKEY ("Insert Key"). From the menu, select "LocRot"
- The positions and rotations of the Objects now have been stored at frame 50.

Press SHIFT+LEFTARROW to move the 'current frame' to 1

- RightMouse-select the 'L' and translate it (GKEY) to the extreme left of the 3DWindow, outside the dashed view borders.
- Press RKEY, and click MiddleMouse, to invoke a two-axis rotation for the Object. Assign the Object a random rotation by moving the mouse diagonally. Press IKEY, ENTER to insert this position.



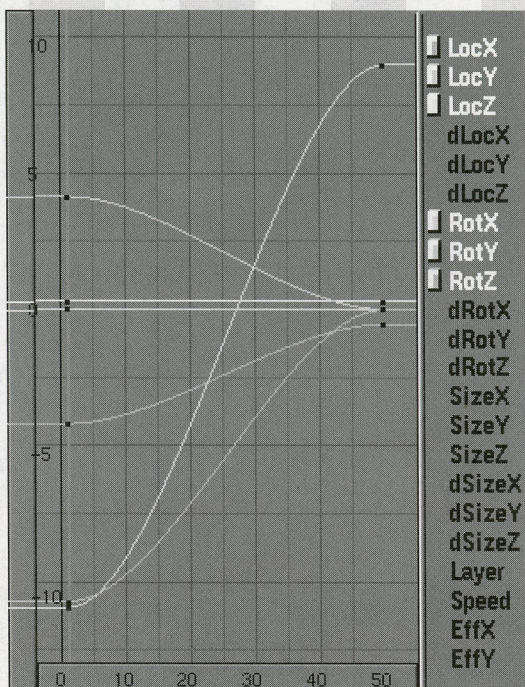
- Repeat these steps for the other three Curve Objects. Try to create a situation such as this:



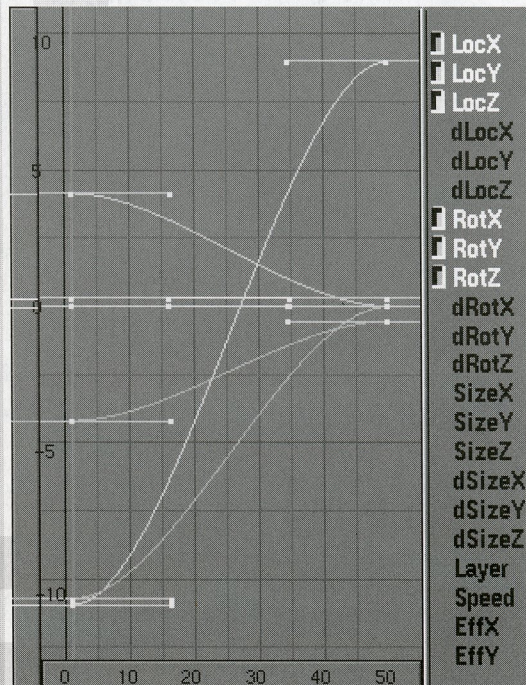
- You can playback this animation with ALT+A (with the mouse cursor in the 3DWindow).
- Press esc to stop playback. If you want to, you can assign other positions or rotations for the Objects by simply inserting keys at frame 1. The old positions then are replaced.
- By default, movement starts and ends smoothly. To change this, we need to edit the animation curves in the so-called 'IpoWindow'. The standard Blender configuration has already included a Screen for this.
- At the top of the Blender screen, you can find this group of buttons.



- Use the small 'menu button' and choose the top item "screen".
- Now you see the first Blender Screen. It has been arranged for animation editing.
- Press HOMEKEY with the mouse cursor in the IpoWindow to view the curves in their entirety.
- To avoid confusing naming conventions, these curves are called the "IpoCurves".
- By RightMouse-selecting each of the Objects, the IpoWindow will automatically display the associated animation curves.



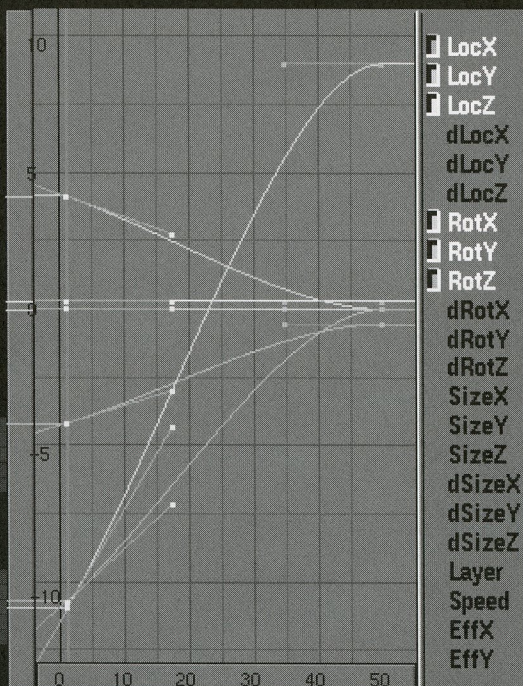
- The IpoCurves are Bezier as well, to see this you have to activate EditMode first:
- Select all IpoCurves, AKEY.
- Press TAB.



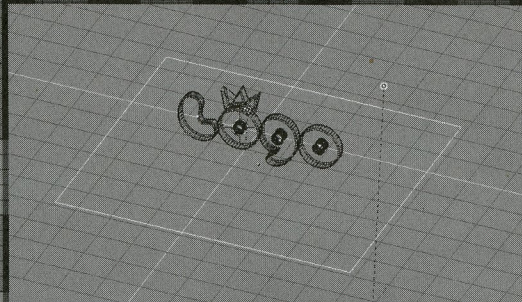
All IpoCurves are now in EditMode. The handles have a yellow colour, indicating 'auto handle' type .

- To make the animation start with an abrupt speed, we change all handles located at frame 1
- Press AKEY to deselect all handles.
- Use BKEY to select all handles at the left side of the IpoCurves.

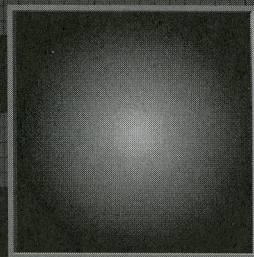
Press VKEY to make the handles 'vector'



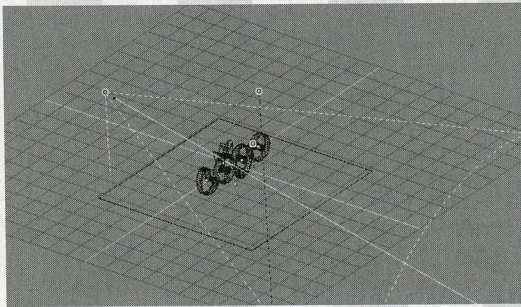
- Do this for the other Curve Objects as well. Then try a playback (ALT+A) to view the animation.
- We are almost finished, we end this tutorial by spicing up the design.
- We add a ground plane for this logo. Enter Top View (PAD_7) and press SHIFT+A, choose "ADD→Mesh→Plane".
- Leave EditMode (TAB) and position (GKEY) the Plane a little under the logo. Scale it 4 times larger using the SKEY.



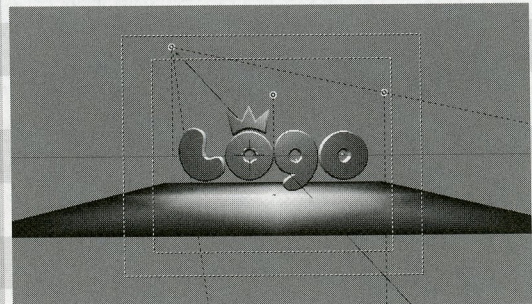
- Press F5 and add a new Material to it. Make it completely black. Set the "Spec" slider at 0.0 to disable the gloss.
- Press F6 and add a new Texture to this Material using the renowned 'menu button' in the header.
- Make the Texture of type "Blend".
- Then press the green "Sphere" option.
- Back to the MaterialButtons (F5), change the purple 'blend' colour to white. You now have created a black plane with a white, circle-shaped blend.



- To make the logo cast shadows on the Plane, we add a third Lamp to the scene: press SHIFT+A, click RightMouse for the main menu, and choose ADD→Lamp.
- Press F4 and make the Lamp a "Spot".
- We don't want this Lamp to influence the current lighting. For this, we activate the LampButtons→OnlyShadow option.
- Now the "Energy" value determines the 'blackness' of the shadow. Set this slider to 0.75.
- Press GKEY to locate the Lamp diagonally behind the logo and use RKEY to point it at the logo.



- Create a quick Gouraud view of this scene by pressing CTRL+Z. At every vertex of the Objects, the color is calculated according to the Materials, Textures and Lamps.
- The Plane is still completely black, because it has only 4 vertices. To have a better resemblance with the final rendering, you can subdivide the Plane a few times: RightMouse-select the Plane and Enter EditMode (TAB), select all its vertices (AKEY) and press EditButtons→Subdivide twice. Leave EditMode again.
- Press PAD_0 to view this:

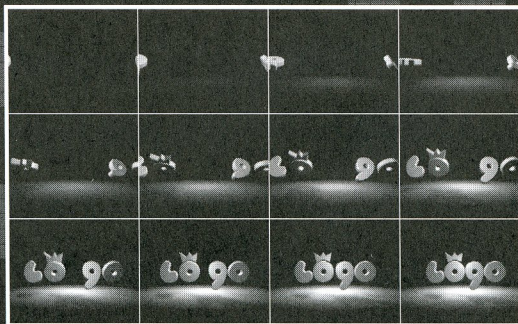


- Press F10, and put on the "Shadows" option.
- Press F12 for a rendering.



- It would be cool to start with a solid black image. For this, we'll add a Material animation for the Plane.
- RightMouse-select the Plane and press F5.
- Go to frame 50: SHIFT+RIGHTARROW.
- With the mouse cursor in the ButtonsWindow, press IKEY.
- This is the "Insert Key" menu for Materials. Select "Alpha".
- Go to frame 1: SHIFT+LEFTARROW.
- Set the "Alpha" slider value at 0.0.
- Press IKEY, ENTER to insert this value at frame 1.
- If you browse the frames (ARROWKEYS) you can see the slider value changing.
- Render frame 1, this should be completely black now.

- For a preview of the rendered animation, you have to adjust a few settings in the DisplayButtons (F10).
- The upper-leftmost button ("Pics") indicates the output directory. Click the button and type a name, press ENTER to leave the button.
- For preview purposes, Blender has the obscure 'HamX' image format. This is a compact 8 bits RLE compressed format. Press the "HamX" button at the right hand side of the buttons menu.
- The image resolution, 320*256, should be fine for realtime playback.
- Activate the "OSA" button for maximum image quality.
- For security, first save your work: press F2, type a filename in the second large textbutton, press ENTER to leave the button, press ENTER again to confirm saving the file.
- Now you are ready to hit the big "ANIM" button.
- Rendering is fast, and should take only a few seconds per image.
- After rendering is finished, press the "PLAY" button to view a realtime playback.
- Use ESC to close the playback window.



More tutorials!

Having more tutorials here would be great fun. Unfortunately, this is beyond the scope of this manual. These first two sections of the manual offer you a quick and accessible first glance at Blender's applications. I hope it evokes enough curiosity and motivation to encourage you to continue mastering Blender.

The next five sections of this manual, sections three to seven, describe all Blender's features, arranged by theme. A lot of practical examples are included there as well. The second half of this manual, sections eight and nine, are for reference purposes.

Many Blender users have already started making tutorials themselves, as HTML pages on the World Wide Web. Links to these tutorials can be found on the Blender site.

used the strategy of

creating this page I used the strategy of
deconstruction, driven by the layering capabilities
of my Mac, this for only one goal, which is to make the
demands on the Blender user (hehe). Therefore I rather call
myself a deconstructionist than a designer. Why design when
I have the opportunity to create chaos? Some say it's a
matter of bad taste. I'd call it the result of caffeine and nicotine:
deconstructionists fuel

layering capab

Part 3

Blender

THE DATA STRUCTURE

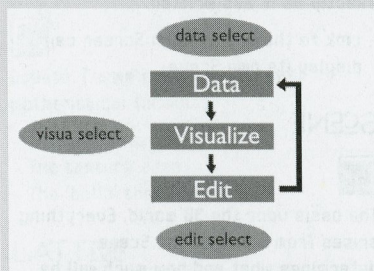
**"WHAT YOU SEE IS WHAT YOU WANT";
"WHAT YOU DO IS WHAT YOU GET!"**

Blender has a strictly organised Object Oriented structure. In designing Blender an attempt was made to define the structures as uniformly as possible, for all the possible 3D representations and for all the universal tools that perform the 3D work.

The structure is such that it allows both users and programmers to work quickly and flexibly; the work method was well suited to developing an in-house 3D design and animation package.

Blender is actually never 'complete' (and that also definitely applies to the version that this book accompanies) and we expect extremely interesting developments for this software.

Although some jokingly call Blender a 'struct' visualizer, the name is actually accurate. The C-programmers among us will certainly understand that. During the first month's of Blender's development, we did little but devise structures and write 'include' files. During the years that followed, only tools and visualisation methods were worked out.



This diagram depicts Blender's basic structure.

DATA SELECT

A user selects 'data'. That means that the user specifies the part of the 3D world he wishes to work on. This is organised in a sort of tree structure. The basic level of Blender's Object Oriented system consists of the Unix/Dos files. These can contain multiple 'scenes' each scene consists in turn of the 3D

objects, materials, animation systems and render settings.

VISUA SELECT

The specified data can be displayed in windows as a 3D wireframe, Zbuffer rendered, as buttons or as a schematic diagram. Blender has an extremely flexible window system for this purpose, a window system that permits any type of non-overlapping layout.

EDIT SELECT

Based on the visualisation selected, the user obtains a number of tools. Editing is always performed directly on the data, not on the visualisation of the data. This seems logical, but this is where a lot of software fails in terms of interactivity and intuition. Partially, it is a speed problem; when visualising in 3D, it can take minutes before the user sees the impact of an edit command. It is also an 'inverse correction' problem: the user is only aware of the visualisation, not of the actual structure that is depicted. This can be quite irritating if you attempt to do something and the results turn out to be the exact opposite of what you intended.

Blender's structure, flexibility and speed are designed to make the program as accessible as possible for the user, but that does not mean the program is simple. Blender has a steep learning

curve; but once you understand the basic structure, you will have a lot of fun and will find the program very easy to use.

THE DATABLOCKS

Blender's DataBlocks are the user's building blocks: they can be copied, modified, and linked to one another as desired.

Creating a link is actually the same as indicating a relationship; this causes a block to be 'used'. This can produce a *right to exist*, or to get a particular *characteristic*.

For example, the block type 'Object' gives a 'Mesh' its right to exist (e.g. a cube). The Object then determines the exact location, rotation and size of the Mesh. More than one Object can have links to the same Mesh.

Each block in Blender has a unique ID, which contains the unique name of the block and, in certain cases, the Library from which the block comes. These IDs allow Blender to arrange the files internally as a sort of file structure; as if it were a collection of directories with files.

This arrangement is also important when combining Blender files or using them as a 'Library' (Libraries are disabled in this version).

Below is a complete overview of the DataBlocks.

MAIN

This is the 'trunk' of the Blender tree structure. All datablocks reside here in lists, arranged by type. The tree structure, which is placed in memory in this way, is virtually identical to the contents of the Blender file. This is the most important reason that reading and writing occurs so quickly.

SCREEN

The complete collection of Screens, Windows and their settings are always saved in a Blender file. Thus, the complete interface is always returned exactly as it was written.

Link to the Scene. Each Screen can display its own Scene.

SCENE



The basis for the 3D world. Everything arises from a Scene. The Scene determines what and how much will be rendered.

Links to Objects. These are special 'Base' links. The selection status and the *layer* of the Object are saved for each link. Different Scenes can have links to the same Objects, but they can be stored in different *layers*.

Link to World (for 'sky' stars, mist).

Link to another Scene. This functions as a 'Set': it is rendered in its entirety.

A list of Sequence strips. For post-processing and (video) montage.

All render information, e.g. the current camera, resolution, output directory, etc.

OBJECT



The universal DataBlock, which is identical for all types of Objects.

Link to ObData, a collective word for Mesh, Curve, Surface, etc. This determines the Object 'type'. Links to a Parent or Tracking Object. Links to Materials.

Link to Ipo (Animation curves).

A list of Effects (e.g. Particles).

The location, rotation, size and a lot of other useful values.

WORLD



Links to Textures.

Settings for Sky.

Settings for the Star generator

Settings for mist, ambient and global lighting.

MESH



The basic ObData type, which consists of a collection of triangles and squares.

Links to Materials

Links to a Key and Ipo, for vertex key framing.

Link to another Mesh, for texture coordinates.

The texture area.

The vertices, faces, vertex colours, 'sticky' coordinates.

CURVE



This ObData is used for Curve, Surface and Font Objects.

- Links to Materials
- Links to a Key and Ipo, for vertex key framing.
- Link to a VectorFont.
- Link to a 'bevel' Object.
- Link to a 'text on curve' Object.
- The texture area.
- Lists with the curves or surfaces themselves.

METABALL



ObData. Forms are created based on mathematical formulas.

- Links to Materials.
- The texture area.
- The 'balls' themselves.

LATTICE



ObData. These are used to deform other Objects.

- Links to a Key and Ipo, for vertex key framing.
- The vertices of the Lattice.

KA

ObData, for creating articulated movements and distortions.

- The vertices, Limbs and Skeleton deformation information.

LAMP



ObData.

- Links to Textures.
- Link to Ipo, for animated lamp settings.
- All settings, e.g. for colour and shadow.

CAMERA

ObData.

- All camera settings.

MATERIAL



- Links to Textures.
- Link to Ipo, for animated Materials.
- All settings, e.g. for colour, sheen and the *mapping* of Textures.

OBJECTS AND OBDATA

TEXTURE



These are separate blocks, since they can also be used by a Lamp or a World.

- Links to Images
- A ColorBand structure.
- All settings for all types of textures.

IMAGE

- File name.
- Mip Mapping data.

IPO

The animation curves, these can be applied universally.

- A list of IpoCurves.

KEY

A universal block that can be used to create Vertex key framing.

- A list of vertices blocks.

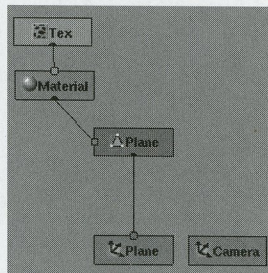
LIBRARY



This block indicates what external Blender file is used as the 'library'.

- A list of all IDs that must be read from the external file.

The user will recognize the Object Oriented structure most clearly when duplicating Objects.



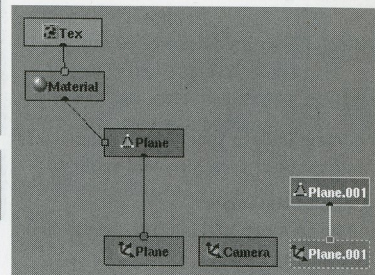
This is, in fact, the basic situation you encounter when Blender starts up; this is the file \$HOME/.B.blend, as included in version 1.5. All examples in this manual assume this start-up file. Users of previous versions are thus strongly advised to perform the complete installation!

You can invoke the DopsWindow from the display in Blender with the command SHIFT+F9. Return to the 3DWindow with SHIFT+F5.

Two Objects are displayed at the bottom; the Camera and the Object with the name "PLANE". The Object "PLANE" has a link to the Mesh, which is also called "PLANE" [only DataBlocks with *different* types can have the same name].

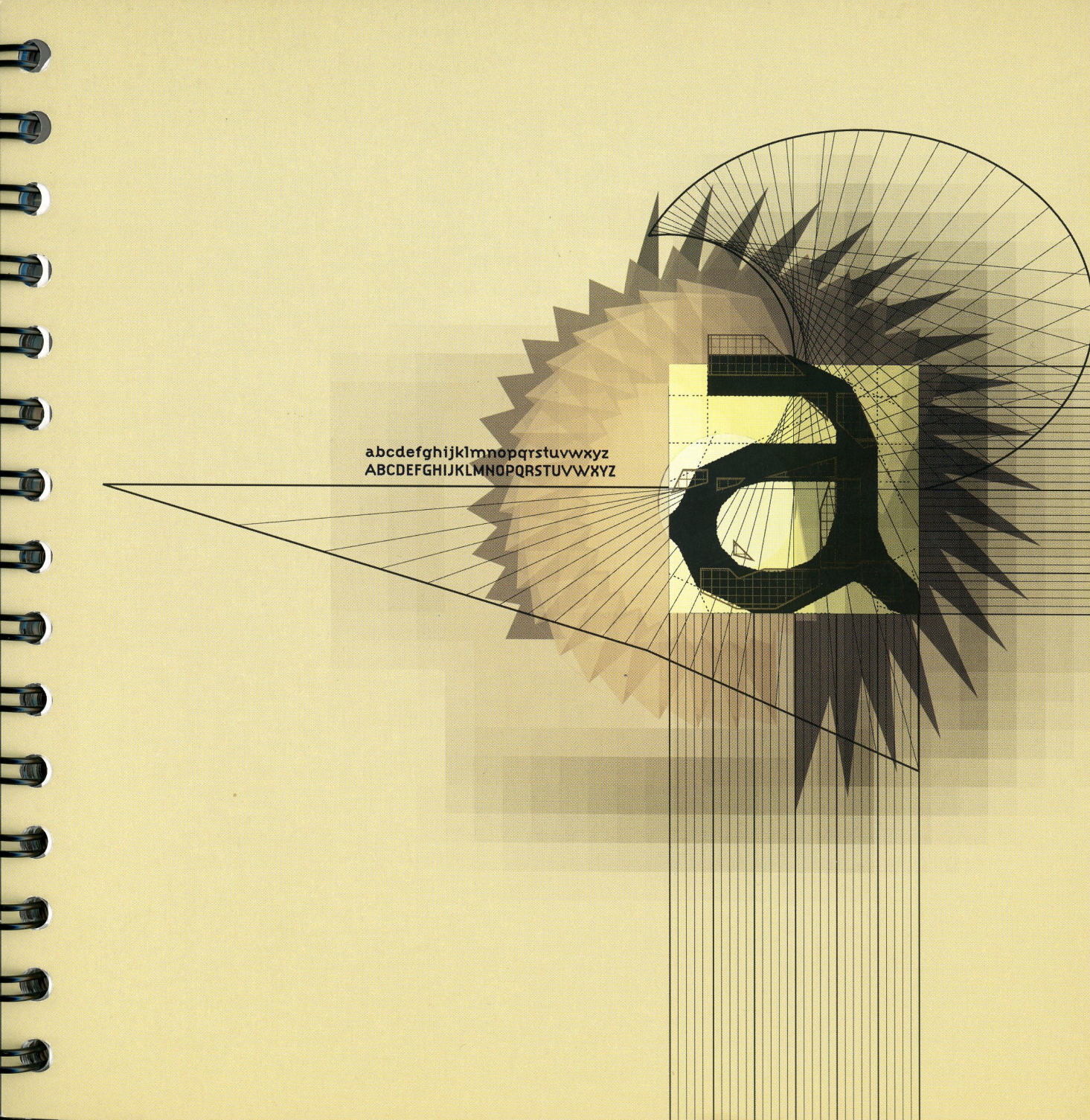
The Mesh block has a link with a Material, and the Material is, in turn, linked to a Texture block.

We are now going to add a new Object. To do this, invoke the SHIFT+A command in a 3Dwindow, and select in this menu the option ADD→Mesh→Plane. Blender goes directly into EditMode for this Object; the individual vertices and faces can be reworked. But we will not do that now; for now, we will leave EditMode using the TAB key.



In the DopsWindow, we now see a new Object and a new Mesh block. The Mesh block still does not have a Material. To assign one, we go to the MaterialButtons [F5]. The buttons window is empty because there is no Material. Create a link to a Material with the small square HeaderButton:

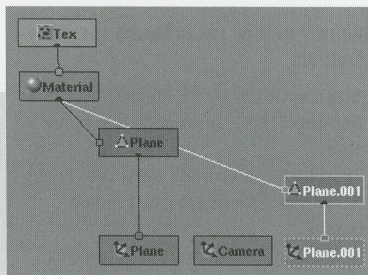
abcdefghijklmnopqrstuvwxyz
ABCDEFGHIJKLMNOPQRSTUVWXYZ





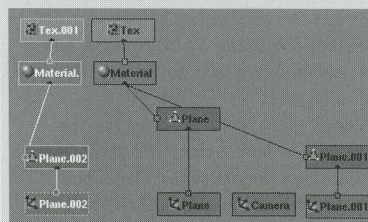


Press this 'MenuBut' to display a list of the available Material blocks. Select the topmost option: "MATERIAL" The buttons now display the Material and a number of other buttons are displayed in the ButtonsHeader. The small button '2' shows that there are two users.



Now we see the new link to the Material displayed in the OpsWindow. This situation shows that both Meshes are using the same Material.

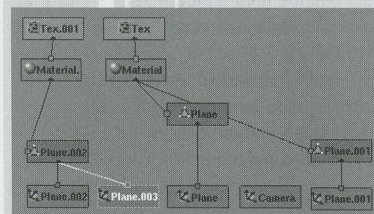
Now we make a copy of the last Object. Use the command SHIFT+D in the 3Dwindow to do this. Blender automatically starts the 'Grabber'; move the mouse and click with LeftMouse to assign the new Object a position.



This command has clearly made a copy of the complete chain
Object→Mesh→Material→Texture.

This is a standard setting. The precise behaviour of the SHIFT+D "DUPLICATE" command can be adjusted by the user to his own taste; for this, read the UserMenu description in the Reference section of this manual.

The alternative to "DUPLICATE" is ALT+D; "LINKED DUPLICATE". If we apply that to the last Object created, we see:



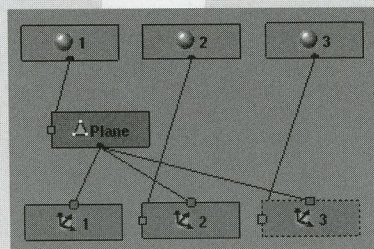
Here we see the power of the Object Oriented system. Extremely complex structures can be created under the full control of the user, with optimum memory usage. At the same time, this makes considerable demands on the user's ability to think abstractly. It is rather easy to loose one's overview, which can unintentionally cause the loss of a great deal of work.

OBJECTS, OBDATA AND MATERIALS.

The Material block is peculiar. During the Blender design phase, the question of whether the material is a feature of an Object or of the ObData arose. Both approaches give logical results and they determine how one must work with Blender.

We finally decided to use the two methods side by side. As an extra concession, a user can specify his personal standard preference in the UserMenu.

A Material can thus be linked to either an Object or a Mesh block. A Material that is linked to an Object takes *precedence* over one linked to a Mesh, allowing a Mesh with multiple links to have different Materials:



In this example, the Objects '1', '2' and '3' all have the same Mesh. Object '1' does not have its own Material, and will thus use the Material of the Mesh. Objects '2' and '3' have their own links to Materials '2' and '3', respectively.

This is a method for creating the same situation:

Start with the basic Blender situation. Use the CTRL+X command to achieve this.

The Object is already selected. Press ALT+D twice to create two linked copies.

In the MaterialButtons [F5], we see these mysterious buttons:

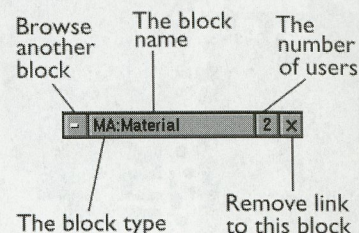


They indicate that the Mesh "Plane" has linked the Material [abbreviated with ME].

Press the 'OB' Button to see the link from the Object to a Material. There is no link here, so the MaterialButtons are no longer displayed and the Header only shows the small square "MenuBut". Now we create a link to a Material. Press the MenuBut in the Header and select the option "ADD NEW". A new Material block is created and linked to the Object.

Perform the same process for the second Object. Select the Object in the 3DWindow (RightMouse). Press the MaterialButtons→OB button and then the ButtonsHeader→MenuBut→"Add New"

Now look at the OpsWindow to make sure that everything went as planned (use the RightMouse-*hold-move* 'Grabber' to move the Ops blocks).



It should now be clear that the primary tool to manage blocks are the DataButtons in the Headers. Virtually every Window type in Blender has these DataButtons. They give systematic access to the complete structure.

CREATING LINKS WITH HOTKEYS

Another frequently-used method for creating links is to use the "MAKE LINKS" menu; CTRL-L.

This copies links from the *active* Object to all *selected* Objects. The menu provides a number of options:

"TO SCENE"

The selected Objects are now linked to another Scene. A second PupMenu asks you to specify the Scene. Objects that appear in more than one Scene have a blue null point.

By linking Objects to multiple Scenes, you can create very efficient structures

without using very much additional memory. The only extra data that are generated are the data for the 'Base' links from the Scene; these contain the selection status and the *layer* of the Object. Multiple linked Objects can be selected independently for each link, and for each link the layer can be changed. If you delete an Object after linking to another Scene, the other link causes that the Object still exists there. This allows you to 'move' an Object to another Scene.

"OBJECT IPO"

All selected Objects get a link to the Object Ipo of the *active* Object. If the active Object does not have an Ipo, the links to Ipo in the selected Objects are erased.

"MESH DATA", or "CURVE DATA", or "FONT DATA", etc.:

All selected Objects get a link to the ObData of the *active* Object. The selected Objects must be of the same type, but Blender catches that.

"MATERIALS"

All selected Objects get identical links, such as the Material(s) of the *active* Object. The complete Material situation is copied correctly. If the active Object has no Materials, all Material links from the selected Objects are erased.

USERS

Blender keeps track of how many users a DataBlock has. This is also important when saving files; only blocks with *users* are saved (this caused headaches during Blender's development for animators who suddenly 'lost' things after a day of work!).

However, *should* used blocks have 'zero' users, their HeaderButtons are displayed in red as a warning. Try temporarily creating an extra link, then save the file and read it again (if necessary with another instance of Blender, to be safe). We believe the system is now watertight... If you have any problems with this, always report them (via E-mail), preferably with enough information to allow us to create the problem again so that we can do something about it immediately.

Not all links to DataBlocks are considered as *users*. These are generally the horizontal links or those that refer back to Objects. These include Parents, TextOnCurve Objects, Bevel Objects and Objects used in Materials for texture coordinates.

It is a good habit to save 'help' Objects in a separate layer (e.g. layer 20).

Another important aspect in the user administration involves deleting objects.

The number of users is then reduced for the *directly* linked blocks. However, this is not true of the underlying structure. For example: When you delete an Object, the linked Mesh block gets 'zero' users, although the Material of the Mesh remains linked. You can link the Mesh back to restore the old situation, as a sort of Undo.

This has a strange effect when saving files: the Mesh is not saved, but the Material is. When you read the file again, the Material appears as a 'zero user' block (possibly with a linked Texture). This is not really a problem, but to get a completely 'cleaned up' file, you will sometimes have to save and re-read a file three consecutive times.

Opinions differ as to how irritating this is. For the time being, Blender will continue to be extremely conservative in releasing blocks.

BLENDER FILE FORMAT

Blender's file format was designed based on the following criteria:

All data are in 1 single file, from the window layout to the complete montage for an animation film lasting several minutes, including all the requisite Scenes and 3D objects.

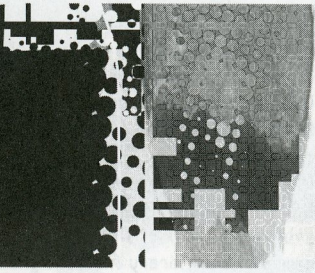
The files must be compact and the program must be able to read and write them extremely fast. In fact, the complete Blender structure is dumped 'as is' in binary format from memory to disk.

The files must be able to use or exchange (parts of) other files, as a sort of library, allowing several users to work together on one project. The files must be able to cope with the continually growing and changing Blender structure. At the very least, newer versions of the program must be able to read old files without problems.

This has resulted in an extremely complicated and obscure file format. The syntax is so strict that a single faulty byte can result in disaster. Each bug in the system is fixed *immediately*, and I can say with confidence that the program is now 99% watertight. The one percent of uncertainty lies in the library system, especially if old and new files are combined. This will be thoroughly tested as soon as the library system is released in a future version.

Another reason, perhaps the most important, that the Blender file format is not easy to release or publish for external use involves the manner in which the internal structure can change 'without problems'.

Each Blender file, *and* each Blender binary, has a sort of 'structure' DNA built in. This is an abstract description of the collection of all the binary data in the file. Each time a file is read, these DNAs are compared to one another. If differences are discovered, Blender tries to make the necessary modifications based on the DNA codes. This is done automatically: all variables can be combined as desired, the 'structs' can be increased or decreased; Blender repairs it all! This makes the file *upward* and *backward* compatible. Even an antique Blender can read the newest files without problems (although this does not happen very often).



The most important performance feature of this system was demonstrated recently when PC files (big endian) and SGI files (little endian) turned out to be exchangeable without causing problems!

In the future, we will decide how to make the files accessible for the users. The obvious solution is to use an external library (for C programmers).

MEMORY

Only DataBlocks with *users* are saved in a Blender file. The 'zero user' blocks remain in memory. The only time Blender frees *all* memory is when files are *read*. Since Blender frees virtually no blocks while working, an enormous number of

'zero user' blocks can accumulate after a few hours. We therefore recommend that you regularly write and re-read files.

AND WHERE IS THE UNDO? BUFFER?

Blender has no true 'undo'. The well-organised structure makes this feasible, but so far we have not found an elegant solution for this.

To compensate for this, the following facilities have been built in:

- A copy of the original ObData is always saved in EditMode. Press the UKEY while in EditMode to retrieve this copy. By regularly pressing TAB-TAB, the user can effectively create a relevant Undo buffer.
- Blender allows you to duplicate a complete Scene. You can experiment with this any way you wish. The results of your experiments can simply be deleted later.
- When Objects are deleted, the DataBlocks linked to them remain. These now have 'zero' users. This allows you to re-link the Meshes or Materials of deleted Objects.

- At a fixed interval, e.g. 3 minutes, Blender can save a 'temp' file. This can be used as a temporary backup or in case of disaster. The file that is written is identical to a Blender file saved in the normal manner. If Blender is in EditMode, then only the original Data are saved, not the data you are working on. In the directory specified in the UserMenu, files are saved under the name <process_id>.blend. This always creates a unique name, so that more than one instance of Blender can write 'temp' files simultaneously to the same machine. When Blender is closed successfully (by the user), the 'temp' file is assigned the name "quit.blend". You can use this file to restore work from an instance of Blender that is closed 'by accident'. Blender saves files very quickly. Normally, you can continue right on working a split second after saving a file of 1-2 Mb.

THE LIBRARIES.

- You can construct a 'history' of Blender files, using the "Versions" option in the UserMenu. This assigns a new version number to an old file when a newer version is saved.

Example:

If Versions is set to '2' and the file 'rt.blend' is saved:

- rt.blend2 is deleted.
- rt.blend1 becomes rt.blend2
- rt.blend becomes rt.blend1
- rt.blend is saved.

Blender allows you to read, append or link (parts of) other files. This option is partially disabled for version 1.5. Still, I would like to describe how it operates.

If you select the 'Load Library' option (SHIFT+F1), the FileSelect appears in a special mode. Blender files are now highlighted as directories. They can also be accessed as directories; they show the internal Blender structure, just as it would appear in a DataView.

Now you can select any number of blocks with RightMouse and add them to the current structure with a simple ENTER. This automatically includes the complete 'underlying' structure: thus, if an Object is selected, the associated lpo, ObData, Materials and Textures are included. If you select a Scene, everything, Objects and all, is added to the current structure.

You can select the manner in which this will occur in the Header of the FileSelect:

APPEND.

This means that the external blocks form a normal part of the current structure, and thus of the file, as well, if a file is written. Selecting 'Append' a second time adds everything, in its entirety, a second time. Since block names must be unique, the names of appended blocks generally differ slightly from the original name (only the number in the name).

LINK.

This is what Libraries are really used for. The specified blocks are now added to the current structure, but Blender

remembers the fact that they are Library blocks. When writing a file, only the name of the Library block is saved and the file from which it was retrieved. This keeps the file quite compact. When the file is read again, Blender reads the original file and then all the Library blocks from the other file(s). This can occur an unlimited number of times, provided the names of the Library files and the blocks remain unchanged.

Blender keeps track of what Library blocks have already been read. Adding the blocks again has no effect.

Blender distinguishes two types of Library blocks: *direct* and *indirect*. The *direct* Library blocks are the ones indicated with the FileSelect. The *indirect* Library blocks are the extra blocks that are included from the underlying structure. Only the *direct* blocks are saved in a file. This enables a Scene linked as a library, even though completely changed in the Library file, to be retrieved without problems, complete with changes, when read.



combined characters

sequence

a sequence
in a certain sequence

sentences combined
sequences are words in a sentence
Words are combined

sentences combined
sequences are words in a sentence
Words are combined

are
are
are



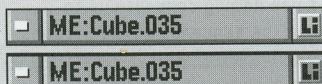
Since Library blocks actually come from other files, they cannot be edited or changed in any way. They are completely static; buttons are blocked. EditMode is no longer available and it is even impossible to move the Object. The only operations permitted are:

Select, view, render
Delete.

Since the selection status and *layer* of Objects in a Scene can be changed, a Library-Object can be selected and you can move it to another layer. Links to Library blocks can be created. For example, a Library Mesh can be used by a local Object.

Make local; unlink, Use the hotkey LKEY, "MAKE LOCAL USER" for this.

Library Object blocks are displayed in Wireframe in blue. They are also displayed clearly in the Headers:



- An orange button means: a *direct* Library block. The block can also be made "LOCAL USER" with this button; the link to the Library file disappears.
- A red button means: an *indirect* Library block. This block does not actually exist, so it cannot be made local.

The "LOCAL USER" operation has two important rules if only part of the linked structure is to be unlinked:

- All *indirect* blocks linked to a previous Library block are converted to *direct* Library blocks.
- A conflict may arise if the Library block must be local, while another Library block has a link to it. This is solved by simply duplicating the Library block.

Working with Libraries requires careful planning and organisation. Changing Library files arbitrarily can cause considerable problems (although generally that only means losing 'data'). You can also go more than one Library level deep: a linked Scene can have Objects from another Library that uses Materials from yet another Library. In practice, it is frequently considered frustrating that Library blocks are static. For situations in which you just want to move an Object or read just a Material, the LKEY has a second option: "MAKE LOCAL ALL".

Libraries work very well in terms of keeping the current file compact. If you work with really large projects, the parts that require the most memory, i.e. large Meshes, or a complete Scene that can function as a 'Set', should be placed in a Library file. Although loading the project will still take just as much time, you will save a lot of time writing it.

THE INTERFACE

SCREENS

Blender's Screens form the basis of the entire interface. The interface was designed based on the idea that overlapping windows, which are common in many programs, do little but make the application confusing.

Blender has a peculiar window system that is based on non-overlapping subdivision of the Screen. Windows are uniformly accessible from the Screen, no matter how large they are. Even buttons can be internally translated or zoomed in or out.

The most important characteristic of the interface is that it is based on full parallel functioning, i.e. the interface cannot block. A variety of visualisations, tools and 'modules' (animation, render, modelling and post-processing) remain permanently available and can be simultaneously used within Blender. They can also be fully configured to meet your personal needs.

A Blender file consists of an unlimited number of Screens. A Screen consists of an unlimited number of Windows and displays the contents of a single Scene.



Use the left-hand "MENUBUT" from the InfoHeader to select a Screen or to create a new Screen. The button with a

cross can be used to delete the Screen. The fact that DataBlocks are always sorted alphabetically in Blender makes it easy to use naming conventions to determine the sequence of the Screens.

Use the hotkeys (ALT-)CTRL+LEFTARROW and (ALT-)CTRL+RIGHTARROW to browse through the Screens.

While in EditMode, a new Screen can be selected only if the Screen in question is associated with the current Scene.

WINDOWS

The Header is a component of all Blender Windows. It contains the most commonly used Blender tools and settings. The button to the far left of the Header always indicates the Window type. Click this button (hold-move) to select a different Window type:



Info. This type provides general information, allows you to select Screens and Scenes and provides access to the UserMenu.

FileSelect. This type allows you to read and write files or to manage the (Unix) file system. Hotkey: F1 of F2.

DataView. Use this type to view Blender's data structure as a file system and to select Objects. Hotkey: SHIFT+F4.

3DWindow. The basis Window for displaying Objects in 3D. Hotkey: SHIFT+F5.

IpoWindow. This Window type visualises animation curves and 'keys' Hotkey: SHIFT+F6.

ButtonsWindow. Allows you to access all of the important blocks in Blender Hotkey: SHIFT+F7

SequenceWindow. The editor used for post-production and (video) editing. Hotkey: SHIFT+F8.

OpsWindow. Displays the Blender structure schematically. Hotkey: SHIFT+F9.

ImageWindow. Displays the Image block. Hotkey: SHIFT+F10.

ImageSelect. Used to visualise and read image files. Hotkey: SHIFT+F11.

When you switch to a different Window, the previous one is not deleted, but can be called up the next time without initialisation in exactly the same form as when you switched even if you save and reload a file in the meantime.

The second button in each Header temporarily toggles the Window to "FULLSCREEN" The hotkey (ALT-)CTRL+UPARROW can also be used for this purpose, as well as to return the Window to its former size.

Another Screen cannot be called up in FullScreen mode.

Place the mouse cursor on the Window edge to change the Window's shape [the cursor will change shape].

Click LeftMouse [hold-move] to move the Window edge.

Click MiddleMouse to split the Window.

Click RightMouse to merge Windows that share a common edge.

A Window becomes active the moment the mouse cursor is positioned over it. Note that key commands and mouse functions can differ from Window to Window.

See the Reference section of this manual for detailed information on working with Screens and the various Window types.

THE BUTTONS

Buttons offer the quickest access to DataBlocks. In fact, the buttons visualise a single DataBlock and are grouped as such. Always use a LeftMouse click to call up Buttons. The Buttons are described below:

Clear

BUT

This button, which is usually displayed in salmon colour, activates a process such as "New" or "Delete".

Shadow

TOGBUT

This button, which displays a given option or setting, can be set to either OFF or ON.

Stick

TOG3BUT

This button can be set to OFF, Positive or Negative. Negative mode is indicated by yellow text.

Glob Orco Stick

ROWBUT

This button is part of a line of buttons. Only one button in the line can be active at once.

ofsX 0.000

NUMBUT

This button, which displays a numerical value, can be used in three ways:

- Hold the button while moving the mouse. Move to the right and upwards to assign a higher value to a variable, to the left and downwards to assign a lower value. Hold CTRL while doing this to change values in steps, or hold SHIFT to achieve finer control.
- Hold the button and click MiddleMouse to change the button to a "TEXTBUT". A cursor appears, indicating that you can now enter a new value. Enter the desired value and press ENTER to assign it to the button. Press ESC to cancel without changing the value.
- Click the left-hand side of the button to decrease the value assigned to the button slightly, or click the right-hand side of the button to increase it.

B 0.800

NUMSLI

Use the slider to change values. The left-hand side of the button functions as a "TEXTBUT"

TE:Tex

TEXTBUT

This button remains active (and blocks the rest of the interface) until you again press LeftMouse, ENTER or ESC. While this button is active, the following hotkeys are available:

- ESC: restores the previous text.
- SHIFT+BACKSPACE: deletes the entire text.
- SHIFT+ARROWLEFT: moves the cursor back to the beginning of the text.
- SHIFT+ARROWRIGHT: moves the cursor to the end of the text.



MENUBUT

This button calls up a PupMenu. Hold LeftMouse while moving the cursor to select an option. If you move the mouse outside of the PupMenu, the old value is restored.



ICONBUT

Button type "BUT" it activates processes.



ICONTOG

Button type "TOGBUT" it toggles between two modes.



ICONROW

As button type "ROWBUT" only one button in the row of buttons can be active at once.



ICONSLI

This button is actually a "ROWBUT" but only the active button is displayed. Hold the button while moving the mouse horizontally to view the available options.



used the strategy of

creating this page I used the strategy of deconstruction, driven by the layering capabilities of my Mac, this for only one goal, which is the pure demands on the Blender user (hehe). Therefore I rather call myself a deconstructionist than a designer. Why design when U have the opportunity to create chaos? Some say it's a matter of bad taste. I'd call it the result of caffeine and nicotine: deconstructionists fuel

Layering capabilities

for only one goal





THE MENUS

TOOLBOX

The Toolbox is invoked with the SPACEBAR or SHIFT+A; in the latter case, the Toolbox appears with the main menu "ADD" active. Almost all key commands are collected in the Toolbox. Normally these are HotKeys for tools that are not included in the interface as buttons.

The left part of the Toolbox contains the main menus. Moving the mouse across this part browses through the contents. The right part shows the specific tool. The key codes are on the extreme right side. The abbreviations have the following significance:

a= AKEY

A = SHIFT+A

c|p = CTRL+P

a|d = ALT+D

ADD	Mesh	>>
VIEW	Curve	>>
EDIT	Surface	>>
OBJECT	Text	
OBJECT	MetaBall	
MESH	Empty	A
CURVE		
KEY	Camera	A
RENDER	Lamp	A
DIV	Ika	A
	Lattice	A

Use ENTER, LeftMouse OR MiddleMouse to activate the tool. Using the ESCKEY or moving the mouse outside the Toolbox causes the Toolbox to disappear.

In fact, the Toolbox generates the specific HotKey; you will therefore see a PupMenu that requests confirmation of the key command in certain cases.

MESH	>Plane	A
VIEW	>Cube	A
EDIT	>Circle	A
OBJECT	>UVsphere	A
OBJECT	>Icosphere	A
MESH	>Cylinder	A
CURVE	>Tube	A
KEY	>Cone	A
RENDER	>	
DIV	>Grid	A
	>	
	>Duplicate	D

If a '>>' is displayed to the right of menu, a submenu is present. Use LeftMouse to view the submenu; use RightMouse to return to the normal menu (the previous images show what happens if ADD→Mesh is selected).

As soon as Blender is in EditMode, the "ADD" submenu is fixed on the particular Object type.

Close the Toolbox and quit EditMode (TAB) before another Object type can be added.

PUPMENUS

OK?

QUIT BLENDER

Many key commands invoke a small requester that asks for confirmation:

- Press ENTER or one of the mouse buttons to confirm the command.
- Press ESCKEY or move the mouse outside the menu and the command is not executed.

Specials

Subdivide

Fractal Subdivide

Smooth

Remove Doubles

Hide

Reveal

Select swap

Flip Normals

Other HotKeys display a list with choices. This type of menu offers the following extra options:

- ArrowUp: the menu jumps up 1 item.
- ArrowDown: the menu jumps down 1 item.
- ONEKEY, TWOKEY, counting from above, the first, second, etc. item is selected.

These PupMenus have a memory. The same HotKey+ENTER repeats the command.

NUMBER MENU

LocX: -7.39

LocY: -3.01

LocZ: 0.00

RotX: 0.96

RotY: -1.01

RotZ: -0.88

SizeX: 1.47

SizeY: 1.47

SizeZ: 1.47

OK

The HotKey NKEY allows you to assign exact numeric values. Depending on the Window, the Editmode, or what is selected, a NumberMenu appears. Change is only activated if ENTER or the "OK" button is pressed. Use ESCKEY or move the mouse outside the menu to exit

the NumberMenu without making any changes.

In the 3DWindow:

Active Object (not in EditMode): The location, rotation (in radians) or size is displayed and can be modified.
EditMode Mesh, Curve, Surface: The location of one selected vertex is displayed.

In the IpoWindow:

Selected vertices of IpoCurves in EditMode: the location of one selected vertex is displayed. If multiple vertices are selected, the movement is applied to all selected vertices.
Selected VertexKey: the Y-position is displayed.





Part 4

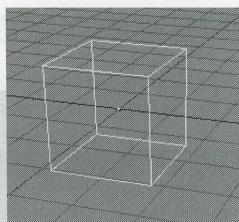
Working with objects

OBJECTS

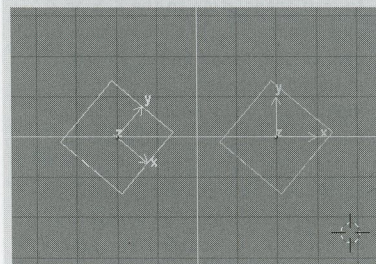
THE OBJECT BLOCK

Objects are the central blocks in Blender's structure. By selecting Objects, you can gain access to the entire linked structure behind the Objects in question: the lpos, ObData, Materials and Textures. Objects 'carry' the other blocks: they give them their right to exist, a place in space or a movement.

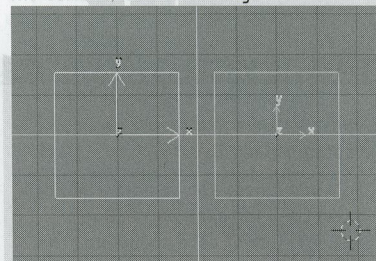
Each Object type has an identical structure. All data shared by arbitrary 3D 'items' are collected in an Object block.



By default, a 'zero point', i.e. the Object's centre, is drawn for each Object. Objects can always be selected at this point. The ObData are also placed here, with a rotation and size that are determined by the Object. For example, this makes it possible for a Mesh to retain its own *local* (texture) space. The determination of exactly what needs to be transformed is therefore extremely important, i.e. should the Object be rotated or should the Mesh itself be rotated (in EditMode)? Although the two options *seem* to have the same results, there is a significant difference:



The same is true of *scaling*:



Two aspects must be considered each time we decide to transform an Object:

- If we change the Object, the Mesh will remain completely unchanged. This can have an impact on the animation curves or relationships with other Objects. The changes in this case are local and *procedural* (i.e. just 9 numbers), which means that we can easily restore the situation prior to the change.
- Changes to the vertices of the mesh affect the texture. This change is *permanent* and is reflected in how all Objects that use the same Mesh are displayed.

In most cases, Meshes can usually be modeled more easily in an *orthogonal* view, without the animations and transformation of the Object. Linking an unrotated and static Object with the same Mesh makes it possible to perform animation and modeling more easily.

SELECTING AND ACTIVATING

Blender makes a distinction between *selected* and *active*.

- Only one Object can be *active* at any given time, for example to allow visualisation of data in buttons. The active *and* selected Object is displayed in a lighter colour than other selected Objects. The name of the active Object is displayed in the InfoHeader.
- A number of Objects can be *selected* at once. Almost all key commands have an effect on *selected* Objects.

A single RightMouse click is sufficient to select and activate an Object. All other Objects (in the visible layers) are then *deselected* in order to eliminate the risk of key commands causing inadvertent damage to those objects. All of the relevant buttons are also drawn anew. Selections can be extended or shrunk using SHIFT+RightMouse. The last Object selected (or deselected!) then becomes the *active* Object.

Use Border Select (BKEY) to more rapidly select a number of Objects at once. None of the Objects selected using this option becomes *active*.

GRAB, ROTATE, SIZE

Most actions on Object involve moving Objects, rotating them or changing their size. Blender offers a wide range of options in these areas. See the 3DWindow section for a comprehensive list. The options are summarised below:

TRANSLATION

GKEY, Grab mode. Move the mouse to translate the Object, then press LeftMouse or ENTER to assign the new location to the Object.

Press ESC to cancel. Translation is always corrected for the view in the 3Dwindow. Use the MiddleMouse toggle to limit translation to the X, Y or Z axis. Blender will then determine which axis to use, based on the already initiated movement.

RightMouse, hold-move. This option allows you to select an Object and *immediately* start Grab mode.

LeftMouse, hold and draw a straight line. This is a 'Gesture' Blender recognises the 'Gesture' and starts Grab mode.

ROTATION

RKEY, Rotation mode. Move the mouse around the rotation centre, then press LeftMouse or ENTER to assign the rotation. Press ESC to cancel. Rotation is always perpendicular to the view of the 3Dwindow.

LeftMouse, hold and draw a curved line. This Gesture starts Rotation mode.

SCALING

SKEY, Scaling mode. Move the mouse from the rotation centre outwards, then press LeftMouse or ENTER to assign the scaling. Use the MiddleMouse toggle to limit scaling to the X, Y or Z axis. Blender determines the appropriate axis based on the direction of the movement.

LeftMouse, hold and draw a sharp corner i.e. a V-shaped line. This is a Gesture, which automatically starts Scaling mode.



EDITMODE

Normally, while working with Objects, you can't change the number of faces or translate some of its vertices. For this, you have to start EditMode by pressing **TAB**.

Now editing is locked at the specific *ObData* block of the *active* Object, e.g. the Mesh, Curve or Surface. A new set of tools now becomes available. You can add vertices and faces, extrude polygons, draw curves or assign colors. You can't add or activate other Objects in EditMode. For this you have to leave EditMode by pressing **TAB** again.

HIERARCHIES

Objects can be linked to each other in hierarchical groups. The Parent Object in such groups passes its transformations through to the Child Objects. This type of hierarchical structure can be very useful. When constructing an imaginary solar system, for instance, the sun rotates the earth and the moon is rotated by the earth. We can even add a wagon with spinning wheels that is rotated by the moon!

Although working with hierarchies is very simple, it does require careful planning. Before creating the structure, the depth of the hierarchy must be determined. Removing an erroneously placed part of the hierarchy and placing it in a new position elsewhere can be very time-consuming and difficult. Working in the wrong direction can also be very difficult. Always start by creating the hierarchy, then animate the objects!

Use the command **CTRL+P** to make the *active* Object a Parent Object for all other selected Objects. When the command is invoked, the *inverse* of the Parent transformation is automatically calculated and stored in the Child Object.

The *inverse* makes it appear to the naked eye as if nothing changed *after* the "MAKE PARENT" command was executed. As an alternative, **SHIFT-CTRL+P**, which invokes the command "MAKE PARENT WITHOUT TRANSFORM", can be used. This causes an 'as is' transformation. Although this results in mathematically logical changes, the results can sometimes be rather unexpected!

Depending on the type of Object, you may also wish to define special Parent relationships:

MESHES, CURVES AND SURFACES: VERTEX PARENT.

Select 1 or 3 vertices in EditMode. Press **CTRL+P**, "MAKE VERTEX PARENT". The Object becomes the Vertex Parent of the selected Objects.

If a single vertex is selected, only the *location* of this vertex is used to perform the Parent transformation. The rotation and size of the Parent are ignored.

If three vertices are selected, the Parent relationship is a normal one and all three of the vertices are used to determine the rotation and location of the Child Object. This method gives interesting effects with Vertex Keys. Objects can be selected in EditMode using **CTRL+Right Mouse**.

MESHES: VERTEX DUPLICATORS.

A Mesh can create a copy of its Child Objects for each vertex, on the vertex. The copies are *procedural*, not 'real'. They are generated for each frame anew. Activate this option using AnimButtons→Duplverts.

LATTICES: DEFORMATIONS.

Using Lattices as Parent adds deformation to Child Objects, in addition to the standard transformations. First, however, the Vertices from the Lattice must be moved! The deformation, which is *procedural*, is repeated for every frame.

IKAS: VERTEX PARENT, LIMB PARENT AND SKELETON DEFORMATION.

If the active Object is an Ika, three options are available via the "MAKE PARENT" command:

"USE VERTEX": one of the vertices from the Ika *chain* becomes the Parent. Only the location transformation is passed on to the Child Objects. Use the NumberButton, which appears automatically, to specify the appropriate vertex.

"USE LIMB": one of the Limbs of the Ika becomes the Parent. Use the NumberButton to specify the appropriate Limb.

"USE SKELETON": all of the Ikas and Emptys that form the Skeleton become the Parent and can deform the Child

Objects. Deformation occurs only in Objects that are *directly* parented to the Skeleton.

Create a Skeleton by selecting one or more Ikas and Emptys, then press CTRL+L. "MAKE SKELETON" The active Ika Object then has the Skeleton information.

TOOLBOX

ADD	Clear size	a/s
VIEW	Clear rotation	a/r
EDIT	Clear location	a/g
OBJECT	Clear origin	a/o
OBJECT	Make Parent	c/p
MESH	Clear Parent	a/p
CURVE	Make Track	c/t
KEY	Clear Track	a/t
RENDER		
DIV		
	Image Displast	c/d
	Image Aspect	a/v

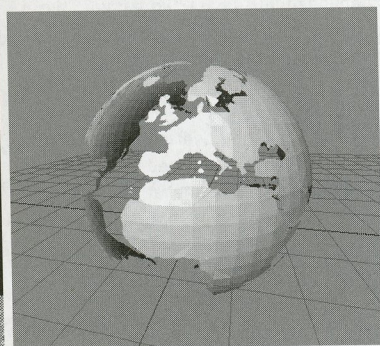
The Toolbox, which can be invoked by pressing the SPACEBAR, displays most of the key commands available to you in Blender. The commands can also be invoked from within the Toolbox.

Use the ADD menu from the Toolbox to create new Objects. If possible, Blender starts EditMode. You will need to quit EditMode using the TABKEY before creating other Objects. Use the hotkey SHIFT+A to invoke the Toolbox displaying the Add menu.

The built-in *primitives* in the ADD menu are designed for use as a convenient point of departure. Users often delete them immediately and then start experimenting with the entire range of options available for the Object types.

THE MESH

THE MESH BLOCK



The Mesh DataBlock consists of vertices, triangles and rectangles. It is extremely easy to build 3D objects using these basis elements and these elements offer extremely precise control over form. In certain cases, it may be advisable to model an Object from a higher 'order' first, e.g. a Curve or Surface, that can be converted to a Mesh for further processing.

Mesheres are displayed with the *transformations* of an Object, with an appearance that is determined based on a Material.

A Mesh can also be linked to a number of Materials, which can be accessed by the *faces* using a simple sequential number, i.e. the Material Index.

The Mesh also saves a separate texture space, which normalises the coordinates of the vertices. By default, the texture space is identical to the *boundingbox* of all of the vertices. The user can specify a texture space using the command TKEY (not in EditMode).

The following data are saved for each vertex:

- A local 3D coordinate.
- A vertex normal (the weighted average of the surrounding face normals).
- A colour (optional).
- A 'sticky' texture coordinate (optional)

The following data are saved for each face:

- 2, 3 or 4 references to a vertex.
- A Material Index number.

Face normals are not saved for each face, but are recalculated each time the Object is rendered.

Vertices and faces can eat up a great deal of the total memory used, which is why they have been configured to take up as little memory as possible.

The following limitations should be kept in mind while using the application:

- The maximum number of vertices in a Mesh is 65535. Additional vertices are permitted in EditMode, but the application generates a warning and blocks EditMode until the command PKEY ["SEPARATE"] is used to place some of the vertices and faces in separate Mesheres.
- The vertex normals are *not* saved in the faces. Although it seems logical, it does make it impossible to achieve 'automatic' *smoothing*.

EDITMODE

Smoothing, which is the use of interpolated vertex normals within a face, can only be controlled in Blender by ensuring that faces share, or do not share, vertices. Links between faces can be deleted using the command `YKEY` ["SPLIT"].

Example: we construct a cylinder with *smoothly* rendered sides, without smoothing the ends.

- SHIFT+A, select ADD→Mesh→Cylinder. This is a standard primitive with vertices linked to each other. You can test the links using the command `LKEY` [with the mouse cursor in the vicinity of a vertex].
- Select EditButtons→Smooth to *smooth* all selected faces.
- Deselect everything: `AKEY`.
- Select one of the ends of the cylinder using `BKEY` [border select].
- Press `YKEY`. The selected part is now unlinked. Test this using the Grabber [`GKEY`, move the mouse cursor, `ESC`].
- Deselect everything again: `AKEY`.
- Select the other end of the cylinder using `BKEY`.
- Press `YKEY`.
- Quit EditMode, hang up a Lamp and render the result [`F12`].

Blender makes use of hotkeys to allow you to work very quickly - using both hands. The corresponding buttons are also available in the ToolBox for alternative use until you have memorised the hotkeys. The example above can also be executed using only the SPACEBAR and the mouse.

When EditMode is started, the vertices and faces in a Mesh are changed into lists of vertices, faces and edges. Each edge forms a link between two vertices and appears only once in the list. The application keeps edges from being duplicated in the list. This makes it possible to use all of the available tools consistently with Meshes.

Although working with lists (a 'C' term) offers the benefit of a highly flexible structure, it can also slow things down in situations in which thousands and thousands of edges and faces have to be managed and compared to one another. In such situations, the display slows down and making the appropriate selections becomes confusing. This is why the application still limits Meshes to 65535 vertices each.

Select an *edge* by selecting both of its vertices. Select a *face* by selecting all three, or four of its vertices.

EXTRUDE

The most important tool for working with Meshes is the "EXTRUDE" command [`EKEY`]. This command allows you to create cubes from rectangles and cylinders from circles. Extrude allows you create things such as tree limbs easily.

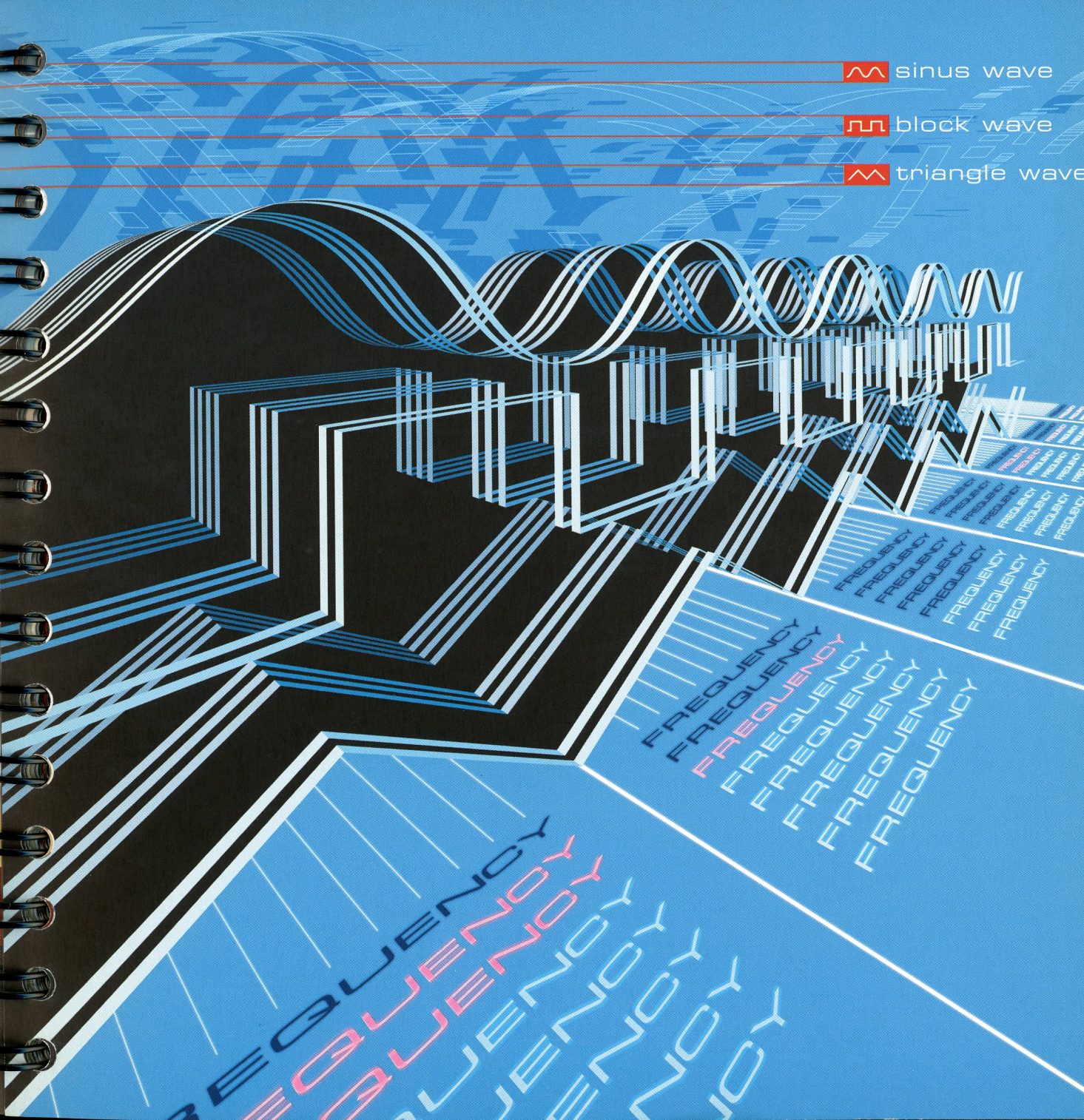
Although the process is quite intuitive, the principle behind Extrude is outlined below for your information:

First, the application determines the outside 'edge' of the Extrude, i.e. which selected edges are allowed to be changed into faces? By default, the application ignores edges with two or more selected faces.

The edges in question are then changed into faces.

If the edges in question had only one face, *all* of the selected faces are duplicated and linked to the newly created faces, e.g. rectangles result in cubes during this stage.

In other cases, the selected faces *themselves* are linked to the newly created faces. This prevents undesired faces from being retained 'inside' if Extrude is invoked again later.



sinus wave

block wave

triangle wave

FREQUENCY
FREQUENCY
FREQUENCY
FREQUENCY
FREQUENCY
FREQUENCY
FREQUENCY
FREQUENCY

FREQUENCY
FREQUENCY
FREQUENCY
FREQUENCY
FREQUENCY
FREQUENCY
FREQUENCY

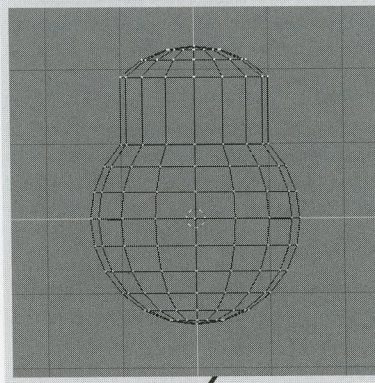


This distinction is extremely important, e.g. during smoothing. The distinction ensures the construction of consistently coherent, closed volumes at all times when using Extrude. Edges without selected faces are simply Extruded. Vertices without selected edges are turned into edges.

Grab mode is automatically started when Extrude is invoked.

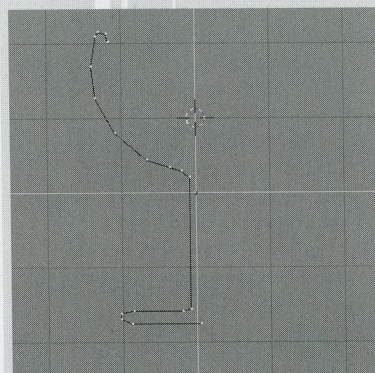
The principle can be tested quite easily by selecting half of a sphere and extruding it. Exactly what you were looking for, is it not?

If you split the selected half of the sphere in this example using **V**KEY before extruding it, a new volume is created.



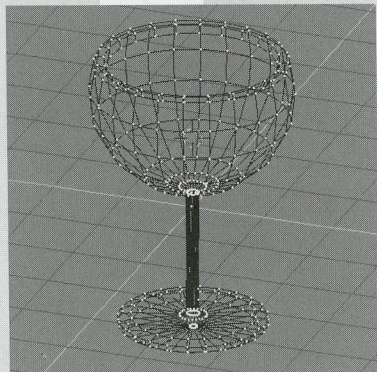
SPIN

A large number of industrial objects are created using a lathe: these are the *surfaces of revolution*. We can illustrate the principle by constructing a wine glass:



- We start — in *front view* [PAD.1] — by drawing a half polygon using the **CTRL+LeftMouse** command, which creates vertices and, if another selected vertex is available, an edge. The polygon can be drawn quite easily using a series of **CTRL+LeftMouse** clicks.
- Select all of the vertices of the polygon [**B**KEY, draw a border using **LeftMouse**].

- Place the 3DCursor on the spot at which you want the revolution axis to occur.
- Now go to *top view* [PAD_7]. The Spin command is always executed perpendicular to the screen.
- Go to the EditButtons [F9] and set the value of "Degr" to 360 degrees and the value of "STEPS" to 24.
- Click the "SPIN" button. If more 3Dwindows are open, you will be prompted to indicate which of them you want to execute the Spin in.



Now we have our wine glass. All we need to do now is delete the 'double' vertices by selecting the entire glass and selecting EditButtons→RemDoubles. Then set all of the faces to *smooth*, quit EditMode, hang up a Lamp and render!

ASSIGNING MATERIALS

By default, faces use the first Material of the Mesh, i.e. *index* number '1'. In order to enable use of more than one Material in each Mesh, the following options are used in the EditButtons:

1 MAT 1 (NumBut)



This button can be used to indicate what Material must be displayed in the buttons, i.e. what Material is *active*. The first number indicates how many Materials are available, the second number indicates the number of the *active* Material.

Each *face* in a Mesh has a corresponding number: the 'Material index'. The same is true of Curves and Surfaces.

? (But)

In EditMode, this Button indicates what *index* number applies to the selected items, i.e. which Material applies to the selected items.

NEW (But)

Create a new *index*. The current Material is given an extra link. If a Material is not already available, a new Material is created.

DELETE (But)

Delete the current *index*. The link to the current Material is deleted. In the ObData, *index* numbers that are already in use are changed.

SELECT (But)

In EditMode, everything with the current *index* number is selected.

DESELECT (But)

In EditMode, everything with the current *index* number is deselected.

ASSIGN (But)

In EditMode, the current *index* number is assigned to the selected faces.

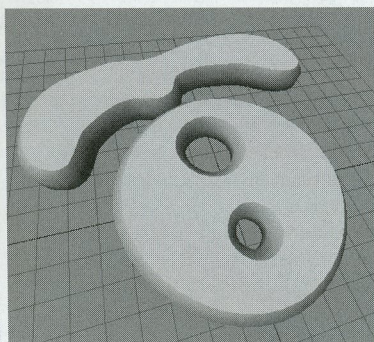
OTHER TOOLS

Where possible, the function of Blender tools and commands is uniform throughout the application. In EditMode, the select (RightMouse), move (GKEY), rotate (RKEY), scale (SKEY) duplicate (SHIFT+D) and delete (XKEY) commands work in the same way as they do when working with Objects.

See "3DWINDOW" and "EDITBUTTONS" in the Reference section of this manual for a comprehensive overview of the tools and options available to you in Blend

CURVE AND SURFACE

THE CURVE BLOCK



The Curve Object can contain an unlimited number of independent curves. With these curves you can create fluid, detailed *procedural* shapes. Each curve is, in fact, nothing more than a series of vertices between which interpolation occurs. Depth (extrude) and *beveling* (the shape of the edges) are built-in facilities that are always simple to modify.

Normally the Curve is 2D. It is displayed with the front and back filled in, with square faces for the sides and beveling. In EditMode, relocation in the Z direction is blocked.

Curves recognize the 'holes' and the beveling direction themselves (toward the outside or the inside). The only thing the user has to do is make sure that the curves do not overlap, because that makes 'filling' problematic.

A Curve is displayed with the *transformations* of an Object and an exterior determined by the Material. Here as well you can have multiple Materials.

In addition, a Curve saves a separate texture space; this normalises the coordinates of the vertices. Normally the texture space is identical to the *boundingbox* of all the vertices. A user-specified texture space can be forced with the TKEY command (outside EditMode).

The following items are saved per Curve (ObData):

- The links to Materials
- Texture space
- Extrude and beveling options
- Type: 2D or 3D
- A list with curves.

The following items are saved per curve (the sequence of vertices):

- Type: Bezier, Nurbs or Poly
- Resolution and *order*
- A Material Index number.

A vertex has:

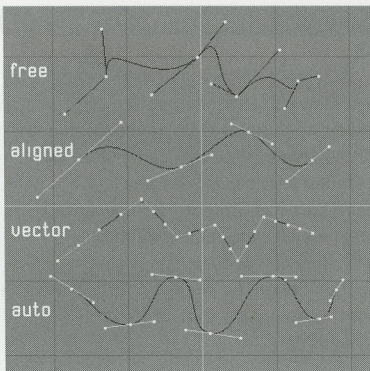
- 3D coordinates
- A weight, for Nurbs curves
- A *tilt*, for 3D curves.

Curves are extremely compact in the memory and in the file. The render faces are only generated at the time of rendering. Because this occurs *procedurally*, it is important to keep the quantity under control.

BEZIERS

Bezier curves are the most commonly used types for designing letters or logos. They are sub-divided into groups of 3 vertices: the *handles*. You can easily change the shape of a curve by moving or rotating handles. Blender distinguishes four types:

- Free Handle (black). This can be used any way you want. Hotkey: HKEY [toggles between Free and Aligned].
- Aligned Handle (pink). These handles always lie in a straight line. Hotkey: HKEY [toggles between Free and Aligned].
- Vector Handle (green). Both parts of a handle always point to the previous handle or the next handle. Hotkey: UKEY.
- Auto Handle (yellow). This handle has a completely automatic length and direction. Hotkey: SHIFT+H.



Handles can be *moved* by first selecting the middle vertex with the RightMouse; this also selects the other two vertices. Now start Grab mode with RightMouse+hold-move.

Handles can be *rotated* by selecting one of the vertices at the end of a handle; again, use the Grabber with RightMouse+hold-move.

As soon as the handles are rotated the type is modified automatically:

Auto Handle becomes Aligned.
Vector Handle becomes Free.

A separate *resolution* can be set for each Bezier curve – the number of points generated between two points in the curve.

NURBS

Nurbs curves have a large set of variables, which allow you to create mathematically pure forms, but working with them requires more intuition:

Knots.

Nurbs curves have *knots*, a row of numbers that specify the precise curve. Two pre-sets are important for this. "UNIFORM" produces a uniform division, for closed curves. "ENDPOINT" sets the knots in such a way that the first and last vertexes are always included.

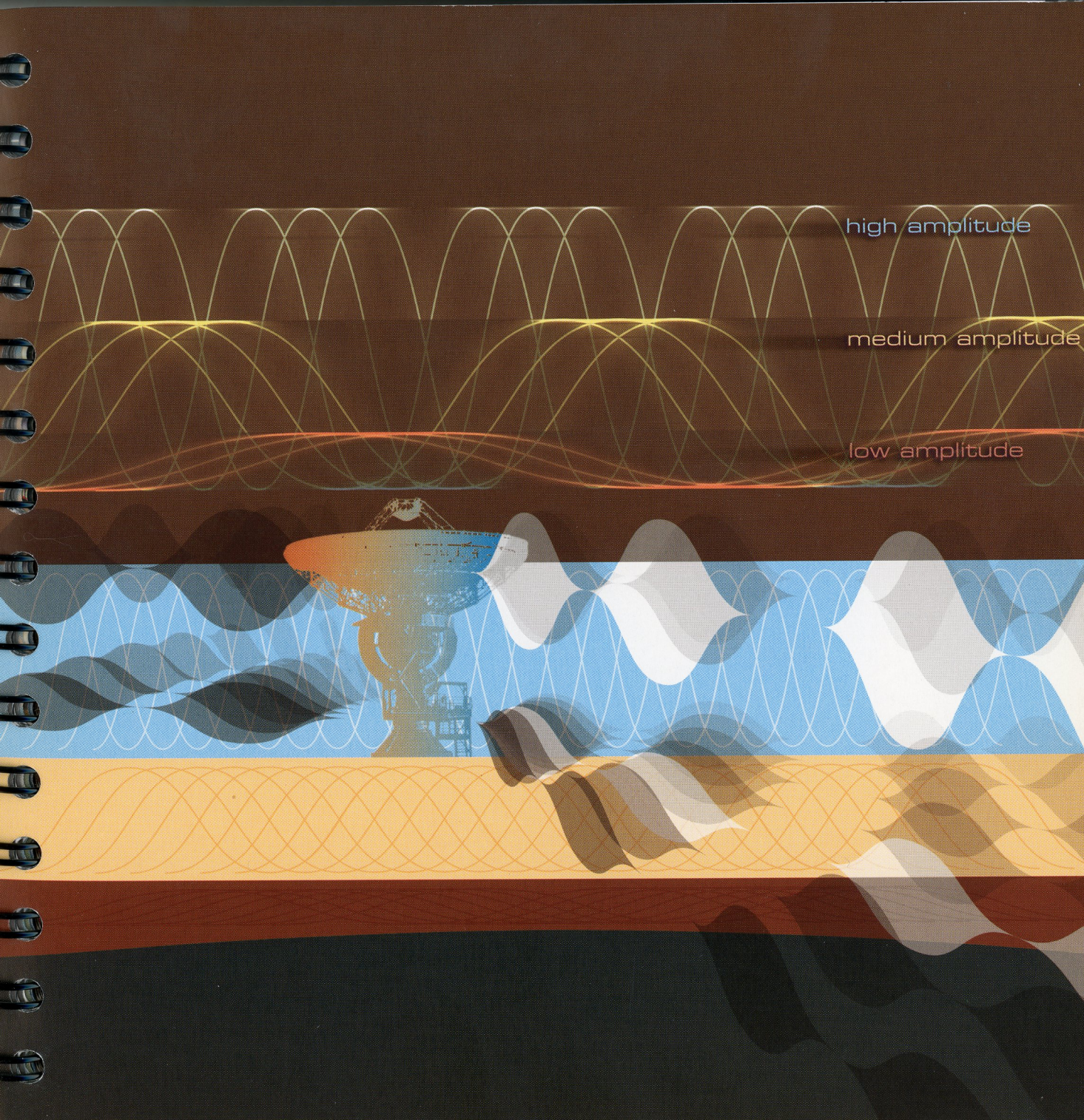
Order

The *order* is the 'depth' of the curve calculation. Order '1' is a point, order '2' is linear, order '3' is quadratic, etc. Always use order '5' for Curve paths; this behaves fluidly under all circumstances, without bothersome discontinuities in the movement.

Weight.

Nurbs curves have a 'weight' per vertex; the extent to which a vertex participates in the interpolation.

Just as with Beziers, the resolution can be set per curve.



high amplitude

medium amplitude

low amplitude



POLY

If no interpolation is needed, you can use the "POLY" type. This has all the facilities, such as *extrude* and *beveling*.

MODELING WITH CURVES

A 'primitive' from the ADD menu is always used as the starting point. By then selecting the ends of the curve, you can add vertices or handles to the curve with CTRL+LeftMouse. Another method for starting a new curve is to duplicate one vertex or one handle from the curve with SHIFT+D.

A curve becomes *active* if one of its vertices is selected with RightMouse. This causes the EditButtons to be re-drawn.

Use the CKEY command to make a curve cyclic, or to undo this action. Only cyclical curves can be rendered 'filled'. Filling closed curves takes place completely automatically. Holes and direction of rotation are recognised automatically. Normally, Blender can fill dozens of curves real-time, but as with everything else, there are limits: it is advisable to split work in different Objects with the PKEY [seParate].

If a Curve is assigned type "3D", vertices and handles can have each 3D coordinate. The automatic front and back filling is then turned off.

As an extra option, an axis rotation ['tilt'] can be specified per vertex. Use the TKEY for this.

The "3D" option is especially interesting for animation paths and for use with "BEVEL OBJECTS".

Blender attempts to create universal tools and commands wherever possible. In EditMode, select (RightMouse), move (GKEY), rotate (RKEY), scale (SKEY) duplicate (SHIFT+D) or delete (XKEY) all work as they do with Objects.

The reference section for "3DWINDOW" and "EDITBUTTONS" contains a complete description of all the available tools and options.

BEVELING

Two pre-sets are included in the EditButtons to realise depth and *beveling*.

Ext1: 0.000
Ext2: 0.000
BevResol: 0

EXT1 (NumBut)

The depth of the extrude.

EXT2 (NumBut)

The depth of the standard bevel; a slanted rounding of the edges.

BEVRESOL (NumBut)

The resolution of the standard bevel; the bevel ultimately becomes a quarter circle.

It is much more interesting if you use *another* Curve to form a 'bevel Object':

BEVOB [TextBut]

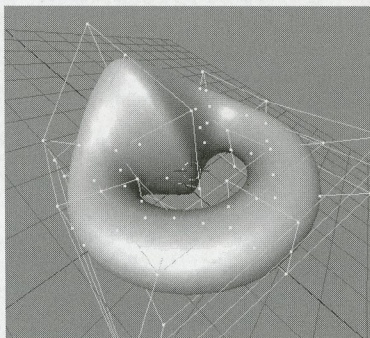
BevOb:

Enter the name of the other Curve Object here. For each interpolated point on the curve, the 'bevel Object' is extruded and rotated.

With this method, you create the rails of a roller coaster with a 3D curve as the basis and two small squares as bevels. Fill in the "RESOLU" values of both Curves carefully, as this beveling can generate many faces.

If the bevels are rendered *smoothly*, the type of Bezier handle on the curve is automatically taken into account. Free handles and Vector handles will always create an 'hard' edge past which is *not* rendered smoothly. This effect can only be seen when rendering, not in the 3DWindow.

THE SURFACE BLOCK



Surfaces are actually an extension of Nurbs Curves. In Blender they are a separate ObData type.

Whereas a curve only produces one-dimensional interpolation, Surfaces have a second extra dimension, called the U and V directions. These allow you to create curved surfaces; a two-dimensional network of vertices defines the form for this.

Use Surfaces to create and revise fluid curved surfaces. They can be cyclical in both directions, allowing you to easily create a 'donut' shape. Surfaces can also be drawn as 'solids' in EditMode [zbuffered, with OpenGL lighting]. This makes working with surfaces quite easy.

One disadvantage of Surfaces in Blender is that the tools provided are somewhat basic in nature, certainly when compared to Blender's big brothers and sisters.

Blender has no tools for creating holes and for melting surfaces, for example. This omission will be corrected in future versions.

Just as with other ObData, a Surface is displayed with the *transformations* of an Object and an exterior determined by the Material. Here, as well, you can have more than one Material.

The following items are saved per Surface (ObData):

- The links to Materials
- Texture space
- The list of surfaces.

The following items are saved per surface (vertices network):

- The dimensions in U and V (in numbers of vertices)
- Resolution and *order* (U and V independent)
- A Material Index number

A vertex has:

- A 3D coordinate
- A weight

MODELING WITH SURFACES

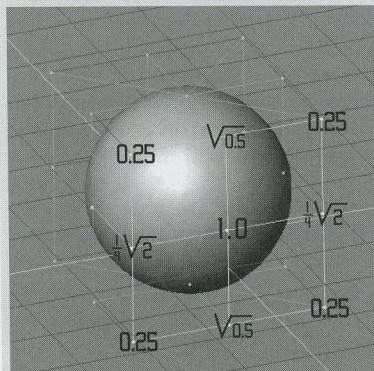
You can take a 'primitive' from the ADD menu as a starting point, or duplicate a single point from an existing surface with SHIFT+D.

Nurbs curves are also a normal part of a Surface; they can be processed and drawn as described above.

Create a surface by extruding an entire curve [EKEY]. Each edge of a surface can then be extruded any way you wish to form the model. Use CKEY to make the U or V direction cyclic. It is important to set the 'knots' to "UNIFORM" or "ENDPOINT" with one of the pre-sets from the EditButtons.

A surface becomes *active* if 1 of its vertices is selected with RightMouse. This causes the EditButtons to be re-drawn.

When working with surfaces, it is handy to always work on a complete column or row of vertices. Blender provides a selection tool for this: SHIFT+R, "SELECT ROW". Starting from the last selected vertex, a complete row of vertices is *extend* selected in the 'U' or 'V' direction. Choose Select Row again with the same vertex and you toggle between the 'U' of 'V' selection.



To create pure circles, globes or cylinders, you must set the weights of the vertices. The example shows this for a globe. Three standard numbers are included as pre-sets in the EditButtons.

Read the weight of a vertex with the NKEY.

A complete overview of all the tools and options is contained in the Reference material for "3DWINDOW" and "EDITBUTTONS".

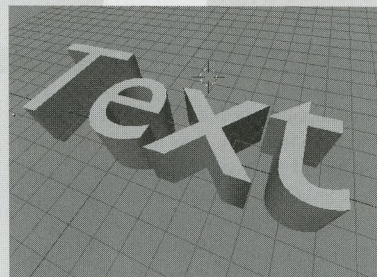
ASSIGNING MATERIALS

As a default, curves and surfaces use the first Material. They have an *index* number of '1'. To be able to use additional Materials, use the options in the EditButtons.

You cannot assign parts of curves or parts of bevels a different Material. You can only do this for a Mesh; to do this, exit EditMode and convert the Object with CTRL+C.

F O N T

THE FONT BLOCK



Font Objects allow you to type and render text in 3D. Blender reads every type of "ADOBETYPE1" format Vector Font with no problem.

By default the Font is 2D. It can be displayed as a Curve, with a filled front and back, and with square faces for the sides and beveling. In fact, a Font block is a 'type of' Curve. Use the convert option **CTRL+C** to edit a Font Object as a Curve.

A Font is displayed with the *transformations* of an Object and an exterior determined by the Material. Here you can only use 1 Material. In addition, a Font saves a separate texture area; this normalises the coordinates of the vertices. Normally the texture space is identical to the *boundingbox* of the text. A texture space specified by the user can be forced with the **TKEY** command (outside EditMode).

The EditButtons contain the settings for outlining, spacing and interlining the text.

Fill in a name for the Curve Object in the button EditButtons→TextOnCurve and the line along which the text is placed will be formed.

ENTERING TEXT

Normally, a Font Object begins with the word "TEXT". This can be deleted simply with **SHIFT+BACKSPACE**.

In EditMode, this Object only reacts to text input. Nearly all of the hotkeys are disabled.

The cursor can be moved with the arrowkeys. Use **SHIFT+arrowkeys** to move the cursor to the end of the lines or to the beginning or end of the text.

Nearly all 'special' characters are available. A summary of these characters follows:

ALT+c: copyright
ALT+f: Dutch Florin
ALT+g: degrees
ALT+l: Brittish Pound
ALT+r: Registered trademark
ALT+s: German S
ALT+x: Multiply symbol
ALT+y: Japanese Yen
ALT+DOTKEY: a circle
ALT+1: a small 1
ALT+2: a small 2
ALT+3: a small 3
ALT+%: promillage
ALT+?: Spanish question mark
ALT+!: Spanish exclamation mark
ALT+>: a double >>
ALT+<: a double <<

Many special characters are, in fact, a combination of two other characters, e.g. the letters with accents. These can be called by first pressing **ALT+BACKSPACE**, and then pressing the desired combination.

Example:

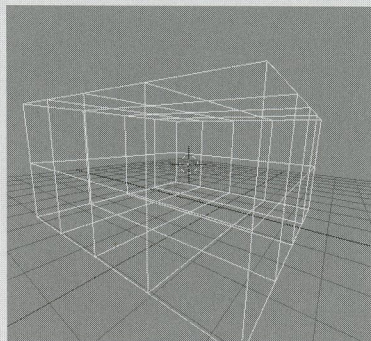
ALT+BACKSPACE, AKEY, TILDE: ã
ALT+BACKSPACE, AKEY, ACCENT: â
ALT+BACKSPACE, AKEY, OKEY: â
ALT+BACKSPACE, EKEY, QUOTE: ë
ALT+BACKSPACE, OKEY, SLASH: ø

This list is too long and too uninteresting to include in its entirety.

Complete ASCII files can also be added to a Text Object. Save this file as "/TMP/CUTBUFFER" and press **ALT+V**.

L A T T I C E

THE LATTICE BLOCK



The Lattice ObData consists of a three-dimensional grid of vertices. If the vertices are moved from their regular positions, this causes a deformation of the Child Objects.

Lattices only affect Meshes, Surfaces and Particles, and can be used to give them a 'Nurbs-like' flexibility.

A Lattice does not affect the texture coordinates of a Mesh or Surface.

MODELING WITH LATTICES

A Lattice always begins as a $2*2*2$ grid of vertices. Use the EditButtons→U,V,W settings to specify the desired resolution.

Then the Lattice can be deformed in EditMode. If there is a Child Object, the deformation is continually displayed and modified.

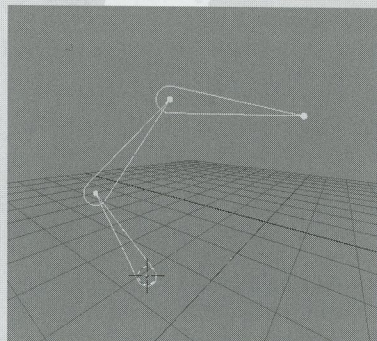
Changing the U,V,W values of a Lattice returns it to a uniform starting position.

Lattices can be used as a modeling tool. They also allow you to make the deformation *permanent*. Use the SHIFT-CTRL+A command. A menu asks: "APPLY LATTICE DEFORM?". The Lattice deformation remains in effect and now works double. So, generally, after 'Applying', the Lattice is unlinked with ALT+P.

You can create interesting animations by moving Child Objects within a Lattice. Lattices deform not only the vertices, but the *movement*, as well. You can create a stream of arrows, for example, that fluidly curve through a Lattice via a given path.

The reference section for "3DWINDOW" and "EDITBUTTONS" contains a complete overview of all the available tools and options.

IKA AND SKELETONS



THE IKA BLOCK

Each Ika block consists of a single *chain* of vertices and limbs. The position of the *chain* is determined by the starting position and by moving the end – the ‘effector’. This makes articulated expression possible, such as arms and legs.

An Ika is further displayed with the *transformations* of an Object. These can only be applied by switching the IKA with TABKEY.

In actuality, the *chain* is 2D, all limbs must always lie in the same plane. If the *effector* moves outside this plane, the *Object* is rotated around the Y axis. Thus, choosing the correct position for the Y axis is absolutely essential for a stable Ika system.

Different Ika (and Empty) Objects can be grouped in a Skeleton. This enables you to assign advanced deformations to Child Objects.

An Ika ObData block saves:

- A starting position. The *chain* has a ‘memory’: it can always return to this position.
- A Skeleton.
- A list of limbs.
- A list of vertices.

A Skeleton saves:

- The list of all Ikas and Empty Objects involved.

A Limb saves:

- A weight that indicates how free the movement is in comparison to other Limbs.
- A transformation matrix for Child Objects.

Ikas in Blender are still at an early stage of development. In particular the ability to apply *constraints* still has to be developed.

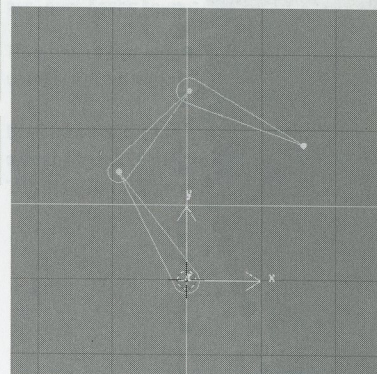
WORKING WITH IKAS

Direct after the ADD command, ‘drawing’ mode is started. The Limbs can now be drawn with LeftMouse. Press MiddleMouse for the last limb or press ESC to terminate drawing.

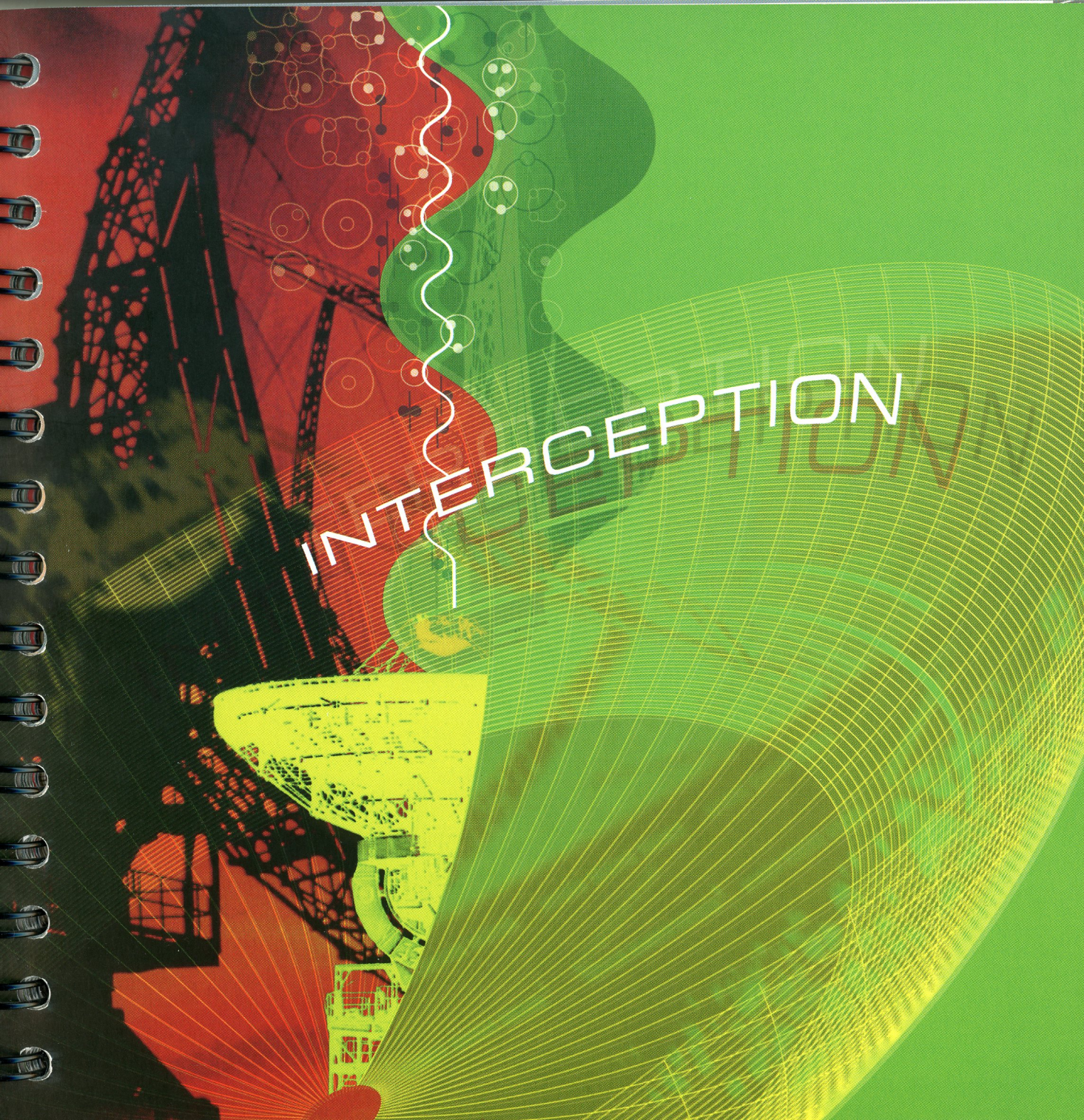
A selected Ika works in Grab mode immediately as an articulated *chain*. This can be turned off with TAB. Now you can simply move or rotate the Ikas.

Different Ikas can be linked to one another in a hierarchy. Here it is important that the basis of the Ika *never* causes movement; only the limbs and the effector cause movement. A basis *can* receive movement, as the Child of other Objects or Ikas.

You must also specify whether the *basis* or the *effector* will affect the Child. A series of menus shows all of the options for the ‘parenting’ of Ikas.



First a simple example. We create an Ika, and parent an ‘Empty’ to the Ika *basis*.



INTERCEPTION



SHIFT+A, select "IKA"

Click three times with the LeftMouse to draw the *chain*. Then press ESC.

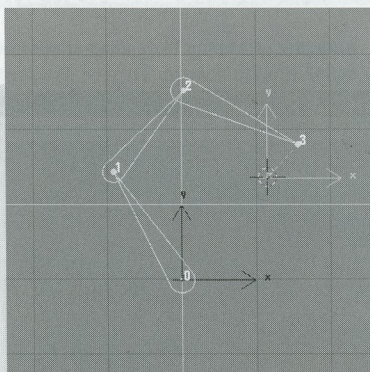
SHIFT+A, select "EMPTY"

Select the Ika with Border Select (BKEY). The Empty remains *active*.

CTRL+P "MAKE PARENT"

If an Ika becomes a Child, a PupMenu always asks: "OK? EFFECTOR AS CHILD" In this case, press ESC.

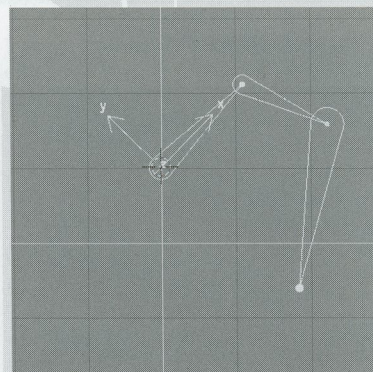
Now move just the Empty to see the effect. Compare this with the effect of only moving the Ika. In fact, you have now created a *chain* with two effectors.



- Now we place a second Empty alongside the Ika (SHIFT+A).
- We select the Ika with SHIFT+RightMouse. This selects both of them, with the Ika *active*.
- CTRL+P now offers three options. Select: "USE VERTEX".
- A button pops up, and the vertex numbers are indicated with the Ika.
- Select '3' from the button.

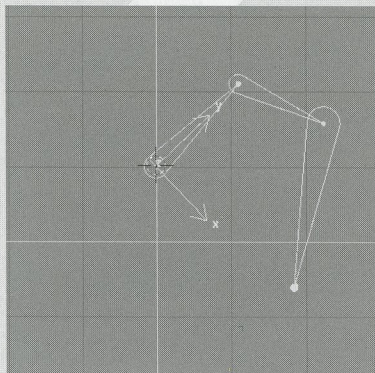
Now move the Ika. We see that the effector only gives the *translation* to the Empty. However, Objects that are parented to a *limb* get a complete transformation.

The manner in which the Emptys affect the Ika in this example is identical to the way Ikas affect each other.



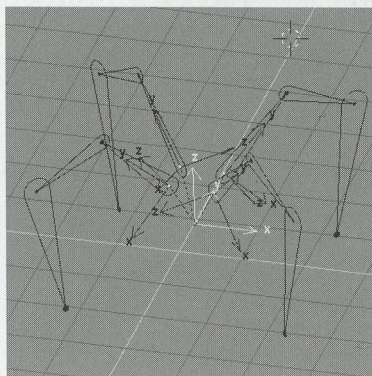
Now we will create a four-legged 'spider'. This examples clearly shows the role of the Ika axis on stability.

- We start, in front view (PAD_1) with a single leg.
- We set the "Axis" as an extra drawing mode in Editbuttons.
- We now see that the Y axis stands perpendicular to the first limb. We can see how much influence this has in the side view (PAD_3). Move the Ika just a bit there (GKEY, ESC.).
- Back in front view we rotate the Ika axis 90 degrees to the right, so that the Y axis points as if it were an extension of the first limb. This can be done with Rotate (RKEY).



- Go back to the side view and test the Ika. This makes a difference! By specifying the axis rotation, we define how much freedom we give the effector. By placing this as far as possible from the Y axis, it has a minimum influence on the Ika's axis rotation.
- Now we go to top view [PAD_7] and temporarily turn off the Ika with TABKEY.
- Now we can rotate the leg a bit and duplicate and rotate three other legs (SHIFT+D).
- In the middle, between the Ika axes, we place an Empty. We parent these to all four legs. Do *not* choose "EFFECTOR AS CHILD".
- Activate the Ikas again with TABKEY.
- Select the Empty with RightMouse and move it. Voila!

SKELETONS



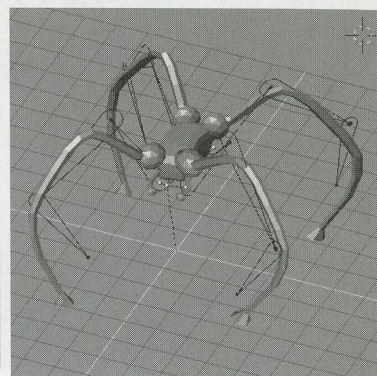
Using the previous example, we can parent a tube to each Limb to create a robot-like creature.

It is much more interesting to use the Ikas to distort a single Mesh, which creates a sort of 'skin' over the skeleton.

To do this, we select the Empty *and* all four Ikas. We make sure that one of the Ikas is *active* and then press CTRL+K (Make skeleton). The colour of the active Ika changes; this indicates that it saves the 'skeleton' information. Emptys can be added to a skeleton to make it more stable and to function as a grasping point.

Each Ika can only have 1 skeleton, but can be used more than once in the skeletons of other Ikas.

Now construct a horrible monster with a Mesh and place it directly over the skeleton.



Make the Ika (with the skeleton) the Parent of the Mesh (CTRL+P), and select the option "USE SKELETON"

You are now finished. Test your creation by moving the legs or by moving the Empty.

THE METABALL BLOCK

MetaBalls consist of spherical elements that can operate on each other's shape. You can only create round and fluid, 'mercurial' or 'clay-like' forms that exist *procedurally*. Use MetaBalls for special effects or as basis for modelling.

In fact, MetaBalls are nothing more than mathematical formulas that perform logical operations on one another [AND, OR], and that can be added and subtracted. This method is also called CSG, Constructive Solid Geometry. Cubes, cylinders, bullets, and surfaces of revolution can also be created, in addition to balls.

Because of its mathematical nature, CSG can be displayed well, and relatively quickly, with Ray Tracing. This is much too slow for interactive displays. Thus *polygonize* routines were developed. The complete CSG area is then divided into a 3D grid, and for each *edge* in the grid a calculation is made whether - and precisely where - the formula has a turning point; a 'vertex' for the *polygonize* is created there.

The available quantity of CSG *primitives* and tools in Blender is limited. This is a point of attention for future Blender versions. The basis is already there; and it is outstandingly implemented.

Unfortunately Blender has little need for modelling systems that are good for Ray

Tracing. However, it is still fun to play with...

A MetaBall is displayed with the *transformations* of an Object and an exterior determined by the Material. Only one Material can be used here.

In addition, MetaBall saves a separate texture area; this normalises the coordinates of the vertices. Normally the texture area is identical to the *boundingbox* of all vertices. The user can force a texture area with the TKEY command [outside EditMode].

The following items are displayed per MetaBall [ObData]:

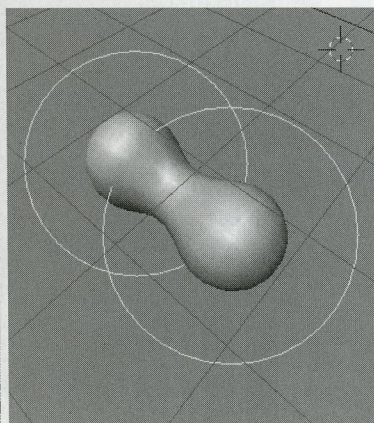
- The link to Material.
- Texture area.
- The resolution of the *polygonize*.
- A list with CSG elements.

The following items are saved per CSG element:

- Size and location
- Type: Sphere or Tube
- The 'stiffness'.

MetaBalls are extremely compact in memory and in the file. The requisite faces are only generated upon rendering. This can take a great deal of calculation time and memory.

M E T A B A L L



MODELING WITH METABALLS

In EditMode you can move and scale the balls or rounded tubes as you wish. This is the best way to construct static - not animated - forms.

MetaBalls can also influence each other *outside* EditMode. Now you have much more freedom in which to work; the balls can now rotate or move; they get every *transformation* of the Parents' Objects. This method requires more calculation time and is thus somewhat slower.

The following rules describe the relation between MetaBall Objects:

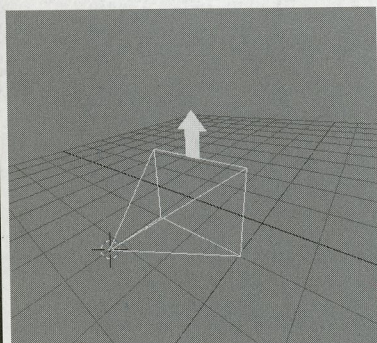
- All MetaBall Objects with the same 'family' name (the name without the number) influence each other. For example "Ball", "Ball.001", "Ball.002", "Ball.135". Note here that we are *not* talking about the name of the MetaBall ObData block.
- The Object with the family name *without* a number determines the basis, the resolution *and* the transformation of the polygonize. It also has the Material and texture area.

To be able to display animated MetaBalls 'stably' it is thus important to determine what Object forms the basis. If the basis moves and the rest remains still, you will see the polygonized faces move 'through' the balls.

The "THRESHHOLD" in EditButtons is an important setting for MetaBalls. You can make the entire system more fluid less detailed or harder using this option. The resolution of polygonize is also specified in EditButtons. This is the big memory consumer: however it is released *immediately after* polygonize. It works efficiently and faster if you work with multiple, more compact 'families' of balls. Because it is slow, the polygonize is not immediately recalculated for each change. It is always recalculated after a Grab, Rotate or Size command.

The Reference part of "3DWindow" and "EditButtons" contains a complete overview of all the available tools and options.

C A M E R A



Blender always renders from the *active* Camera. You can have an unlimited number of Cameras, but only the Camera

you *activate* with CTRL+PAD_0 determines the display. Blender has a right-hand axis system. For a Camera display this means that the Y axis is up, the X axis runs horizontal and and you look in the direction of the negative Z axis.

Each Object in Blender can act as a camera (with CTRL+PAD_0), but only a Camera block provides settings for the lens and clipping. The Spot Lamp is an exception to this. Now the "SPOTSIZE" and shadow buffer limits are correctly displayed. Use this to precisely align a shadow Lamp and to adjust it. The command ALT+PAD_0 always return you to the last *Camera* Object used.

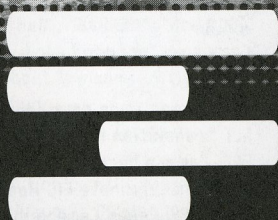
Each 3DWindow can have its 'own' camera, apart from that of the Scene. This camera is *not* rendered.

Use the 'lock' IconBut in the 3Dheader for this. If it is OFF, the active camera and layer data of the 3DWindow are no longer linked to the Scene.

The LocalView option of a 3DWindow can also have its own - and thus temporary - camera.

Active camera's offer a number of extra options for transformations in the 3DWindow. These only work in *camera view*: NUMPAD_0.

- Grab mode: now there is horizontal and vertical translation, from the point of view of the camera. Use the MiddleMouse toggle to zoom in and out.
 - Rotate mode: this allows you to align the camera with the MiddleMouse toggle.
 - Fly mode: (SHIFT+F). The mouse cursor jumps to the middle of the window. The operation is as follows:
 - Mouse cursor movement determines the view direction
 - LeftMouse click [repeated] fly faster
 - MiddleMouse click [repeated] fly slower
 - LeftMouse+MiddleMouse: sets speed to zero.
 - CTRL: translation down (negative Z)
 - ALT: translation up (positive Z)
 - ESC: Camera back to the starting position; terminate Fly Mode.
 - SPACEKEY: Leave Camera in this position. Terminate Fly Mode.
- (Avoid looking straight up or down. This causes irritating turbulence.)



Part 5

Animation with objects

OBJECT IPOS

KEYS AND CURVES

Two methods are normally used in animation software to start a 3D object moving.

- KEY FRAMES.

Complete positions are saved for certain time units [frames]. An animation is created by moving an object through them interpolated fluidly. The advantage of this principle is that it allows you to work with clearly visualised units. The animator works from position to position and can change already created positions or move them in time.

- MOTION CURVES.

Curves can be drawn for each XYZ component for location, rotation and size. These, in fact, form the graphs for the movement, with time set out horizontally and the value set out vertically. The advantage of this system is that you have precise control over the results of the movement.

Both systems are completely integrated in Blender: in the "Ipo".

Fundamentally, the Ipo system consists of the standard motion curves. A simple press of a button changes the Ipo to a key system, without conversion and with no change in the results. The user can work any way he wishes with the keys and switch to motion curves and back again - whatever produces the best result or satisfies the user's preferences.

This chapter first describes the basic principles of the Ipo block and working with motion curves. We close with the "IPOKEY" system.

IPO BLOCK

The Ipo block in Blender is universal. It makes no difference whether an Object's movement is controlled or the Material's settings. If you once learn to work with Object Ipos, how you work with other Ipos is obvious.

Blender does distinguish between different types of Ipos. This only has to do with the type of blocks on which Ipos can work. It is better not to link Object Ipos to a Material! The user does not need to think about this; the interface keeps track of it automatically.

Every type of Ipo block has a fixed number of available channels. These have a name (LocX, SizeZ, enz.) which indicates how they can be applied. When you add an IpoCurve to a channel, animation starts up immediately. At your discretion, and there are separate channels for this, a curve can be linked directly to a value or can only affect a difference from this. The latter enables you to move an Object as usual with the Grabber, while the actual location is determined by IpoCurves relative to it.

The Blender interface offers extensive options for copying Ipos, linking Ipos to more than one Object (one Ipo can animate multiple Objects.), or deleting Ipo links. The IpoWindow Reference section gives a detailed description of this. This chapter is restricted to the options for application.

MAKING IPOs IN THE 3DWINDOW

Insert Key
Loc
Rot
Size
LocRot
LocRotSize
Layer
Avail

The most simple method for creating an Object Ipo is with the IKEY, "INSERT KEY", command in the 3DWindow. A PupMenu provides a wide selection of options. We will select the topmost option: "LOC". Now the current

location, X-Y-Z, is saved. Everything takes place automatically:

- If there is no Ipo block, a new one is created and linked to the Object.
- If there are no IpoCurves in the channels "LOCX", "LOCY" and "LOCZ", these are created.
- Vertices are then added in the IpoCurves with the exact values of the Object location.

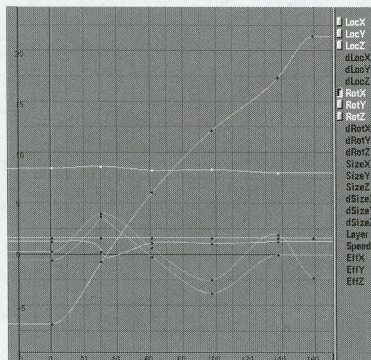
We go 30 frames further (3 x UPARROW) and move the Object. Again we use IKEY and immediately press ENTER. The new position is inserted in the IpoCurves.

We can see this by slowly paging back through the frames (LEFTARROW). The Object moves between the two positions.

In this way, you can create the animation by paging through the frames, position by position. Note that the location of the Object is directly linked to the curves. When you change frames, the Ipos are always re-evaluated and reapplied. You can freely move the Object within the same frame, but since you have changed frame, the Object 'jumps' to the position determined by the Ipo.

The rotation and size of the Object are completely free in this example. They can be changed or animated with the "INSERT KEY"

THE IPOWINDOW



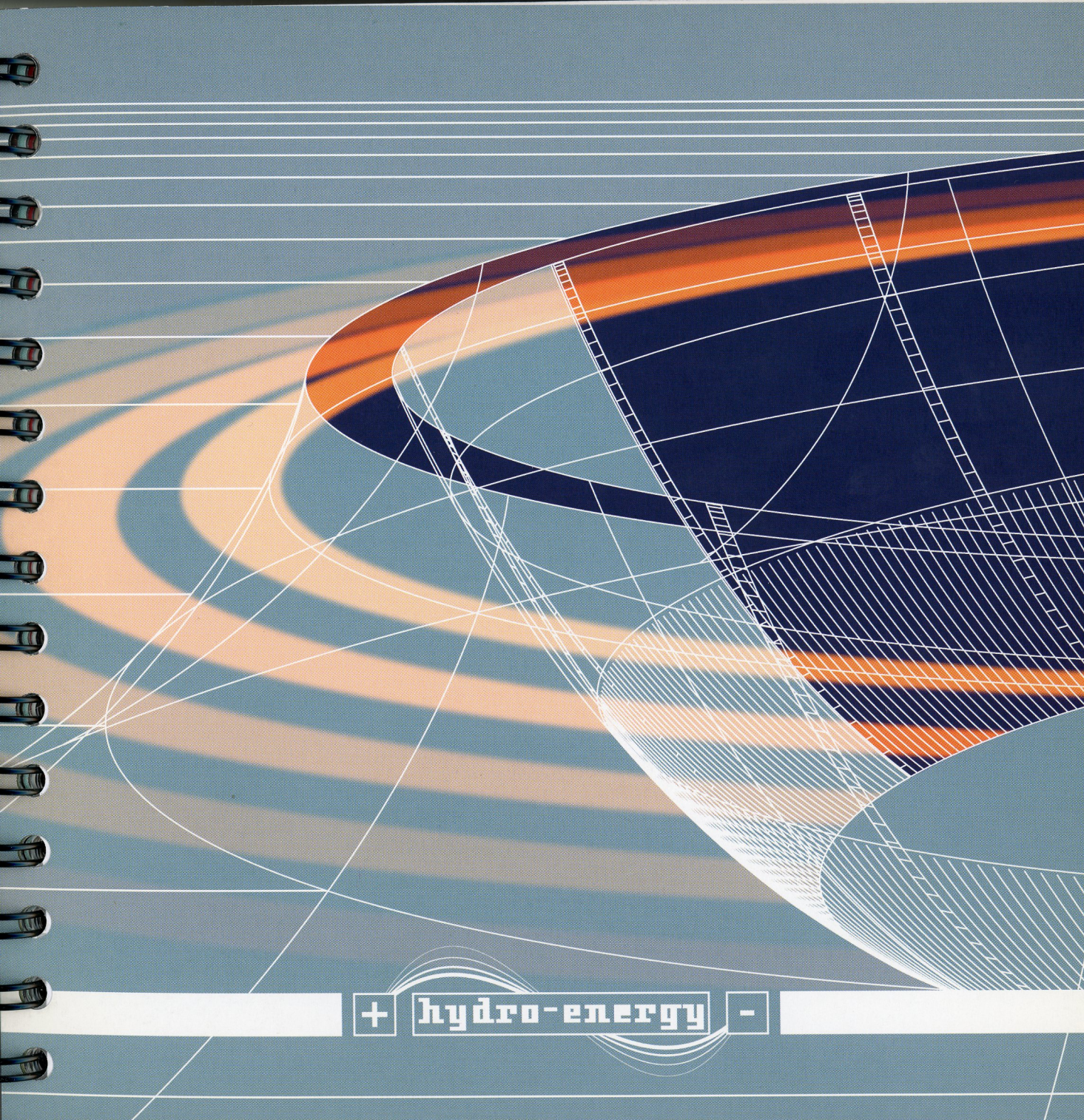
Now we want to see exactly what happened. The first Screen for this is initialised in the standard Blender start-up file. Activate this Screen with (ALT-CTRL+LEFTARROW).

At the right we see the IpoWindow displayed. This shows all the IpoCurves and shows the channels used and those available. You can zoom in on the IpoWindow and translate, just as everywhere else in Blender with (CTRL+) MiddleMouse.

In addition to the standard channels, you have the 'delta' options, such as "DELLOCX". These channels allow you to assign a relative change. This option is primarily used to control multiple Objects with the same Ipo. The 'delta' channels make it also possible to work in animation 'layers'. You can achieve subtle effects this way without having to draw complicated curves.

Each curve can be selected individually: with RightMouse. In addition, the Grabber and Size modes operate here just as in the 3DWindow.

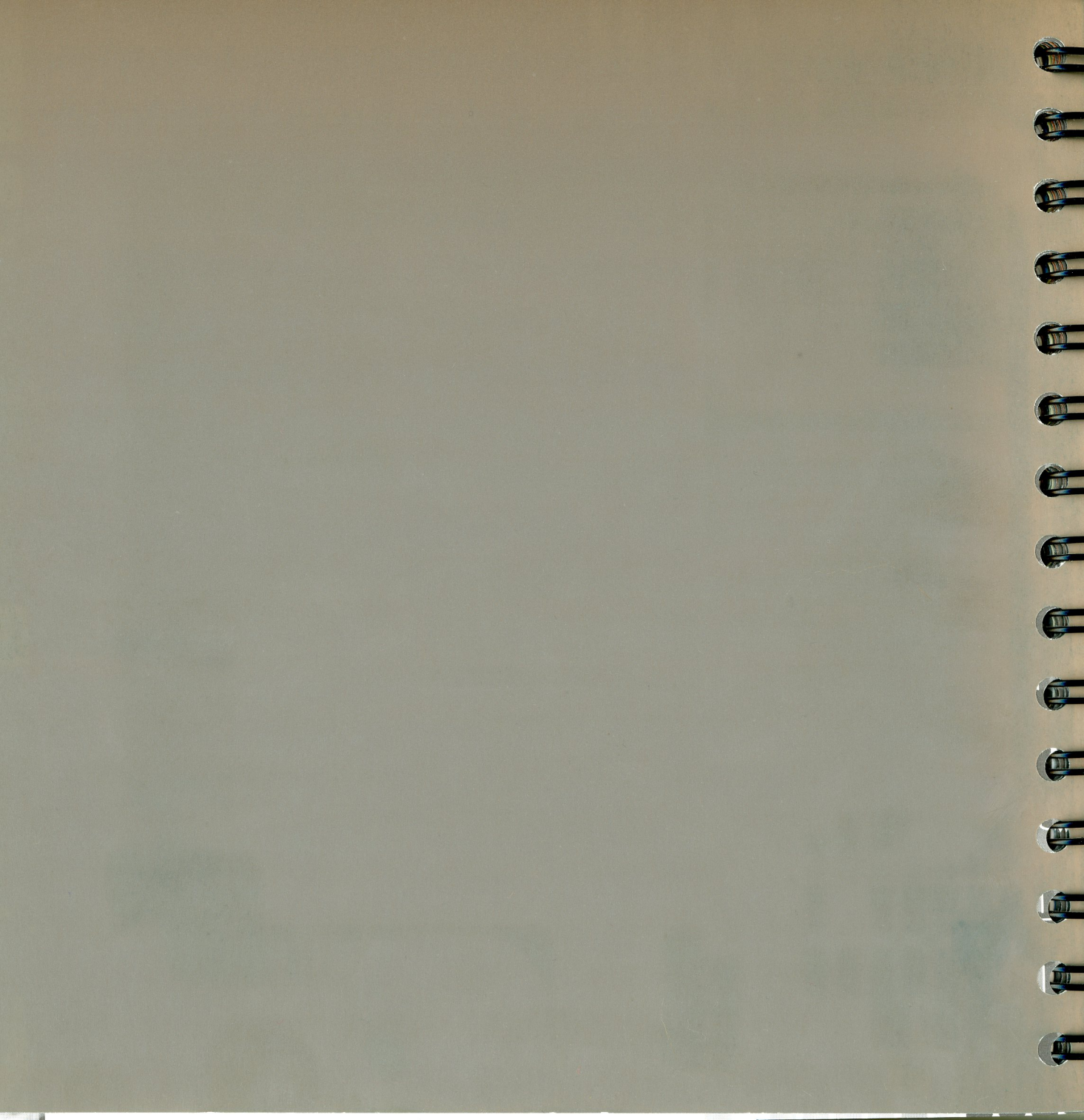
By selecting all curves (AKEY) and moving them to the right (GKEY), you move the complete animation in time.



+

hydro-energy

-



BEZIERs

Each curve can be placed in EditMode individually, or you can do it collectively. Select the curves and press TAB. Now the individual vertices and handles of the curve are displayed. The Bezier handles are coded, just like the Curve Object:

Free Handle [black]. This can be used any way you wish. Hotkey: HKEY (switches between Free and Aligned). Aligned Handle [pink]. This arranges all the handles in a straight line. Hotkey: HKEY (toggles between Free and Aligned).

Vector Handle [green]. Both parts of a handle always point to the previous or next handle. Hotkey: VKEY.

- Auto Handle [yellow]. This handle has a completely automatic length and direction. Hotkey: SHIFT+H.

Handles can be moved by first selecting the middle vertex with RightMouse; this selects the other two vertices as well. Then immediately start Grab mode with RightMouse-hold-move.

Handles can be rotated by selecting of one of the vertices at the end of a handle. Then use the Grabber again with RightMouse-hold-move.

As soon as handles are rotated, the type is changed automatically:

- Auto Handle becomes Aligned.
- Vector Handle becomes Free.

"Auto" handles are placed in a curve by default. The first and last Auto handles always move horizontally, which creates a fluid interpolation.

IPOCURVES

The IpoCurves have an important feature that distinguishes them from normal curves; it is impossible to place more than one curve segment horizontally.

Loops and circles in an Ipo are senseless and ambiguous; an Ipo can only have 1 value at a time. This is automatically detected in the IpoWindow.

By moving part of the IpoCurve horizontally, you see that the selected vertices move 'through' the curve. This allows you to duplicate parts of a curve (SHIFT+D) and to move them to another time frame.

It is also important to specify how an IpoCurve must be read outside the curve itself. There are three options for this in the IpoHeader.



EXTEND MODE CONSTANT (IconBut)

The ends of selected IpoCurves are continuously (=horizontally) extrapolated.

EXTEND MODE DIRECTION (IconBut)

The ends of the selected IpoCurves continue in the direction in which they ended.

EXTEND MODE CYCLIC (IconBut)

The complete width of the IpoCurve is repeated cyclically.

In addition to Beziers, there are two other possible types for lpoCurves. Use the `TKEY` command to select them. A PupMenu asks what type the selected lpoCurves must be:

- "CONSTANT": after each vertex of the curve, this value remains constant. No interpolation takes place.
- "LINEAR": linear interpolation occurs between the vertices.
- "BEZIER": the standard fluid interpolation.

DRAW lpoCURVES

The lpoCurves can also be drawn 'by hand'. Use the `CTRL+LeftMouse` command. Here are the rules:

- There is no lpo block yet (in this window) and one channel is selected: a new lpoBlock is created along with the first lpoCurve with one vertex.
- There is already an lpo block, and a channel is selected without an lpoCurve: a new lpoCurve with one vertex is added
- Otherwise only a new point is added to the selected lpoCurve.
- This is not possible if multiple lpoCurves are selected or in EditMode.

This is the best method for specifying axis rotations quickly. Select the Object. In the lpoWindow, press one of the "ROT" channels and use `CTRL+LeftMouse` to insert two points. If the axis rotation must be continuous, you must use the button lpoHeader→"Extend mode Directional".

ROTATIONS AND SCALING

One disadvantage of working with motion curves is that the freedom of transformations is limited. You can work quite intuitively with motion curves, but only if this can be processed on an XYZ basis. For a location, this is outstanding, but for a size and rotation there are better mathematical descriptions available: matrices [3*3 numbers] for size and quaternions [4 numbers] for rotation. These could also have been processed in the channels, but this can quite easily lead to confusing and mathematically complicated situations.

Limiting the 'size' to the three numbers XYZ is obvious, but this limits it to a rectangular distortion. A diagonal scaling such as 'shearing' is impossible. This can be solved simply by working in hierarchies. A non-uniform scaled Parent will influence the rotation of a Child as a 'shear'

The limitation of the three number XYZ rotation is less intuitive. This so-called Euler rotation is not uniform – the same rotation can be expressed with different numbers – and has the bothersome effect that it is not possible to rotate from any given position to another the famous gimbal lock.

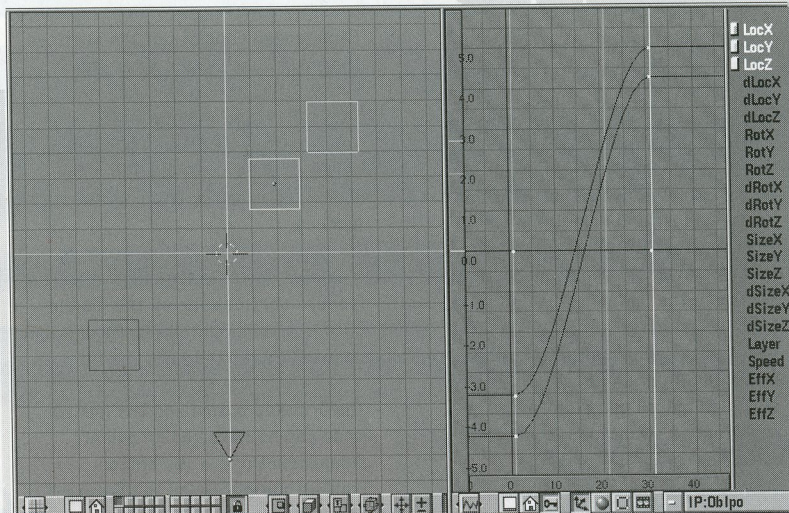
While working with different rotation keys, the user may suddenly be confronted with quite unexpected interpolations, or it may turn out to be impossible to force a particular axis rotation when making manual changes. Here, as well, a better solution is to work with a hierarchy. A Parent will always assign the specified axis rotation to the Child.

[It is handy to know that the X, Y and Z rotations are calculated one after the other. The curve that affects the "ROTX" channel, always determines the X axis rotation].

Luckily Blender calculates everything internally with matrices and quaternions. Hierarchies thus work normally and the Rotate mode does what you would expect. Only the lpos are a limitation here, but in this case the ease of use prevails above the not very intuitive mathematical purity.

IPoKEYS

The easiest way to work with motion curves is to convert them to IpoKeys. We return to the situation in the previous example: we have specified two positions in a Object Ipo in frame 1 and frame 31 with "INSERT KEY". At the right of the screen, you can see an IpoWindow. We set the current frame to 21.



Press **K**KEY while the mouse cursor is in the 3DWindow. Two things happen now:

- The IpoWindow switches to IpoKey mode.
- The selected Object is assigned the "DRAWKEY" option.

The two actions each have separate meanings.

- The IpoWindow now draws vertical lines through all the vertices of all the visible IpoCurves. Vertices with the same 'frame' value are linked to the vertical lines. The vertical lines [the

"IPoKEYS") can be selected, moved or duplicated, just like the vertices in EditMode. You can translate the IpoKeys only horizontally.

- The position of the Object for each IpoKey is drawn in the 3DWindow.

In addition to now being able to visualise the key positions of the Object, you can also modify them in the 3DWindow. In this example, use the Grab mode on the Object to change the selected IpoKeys.

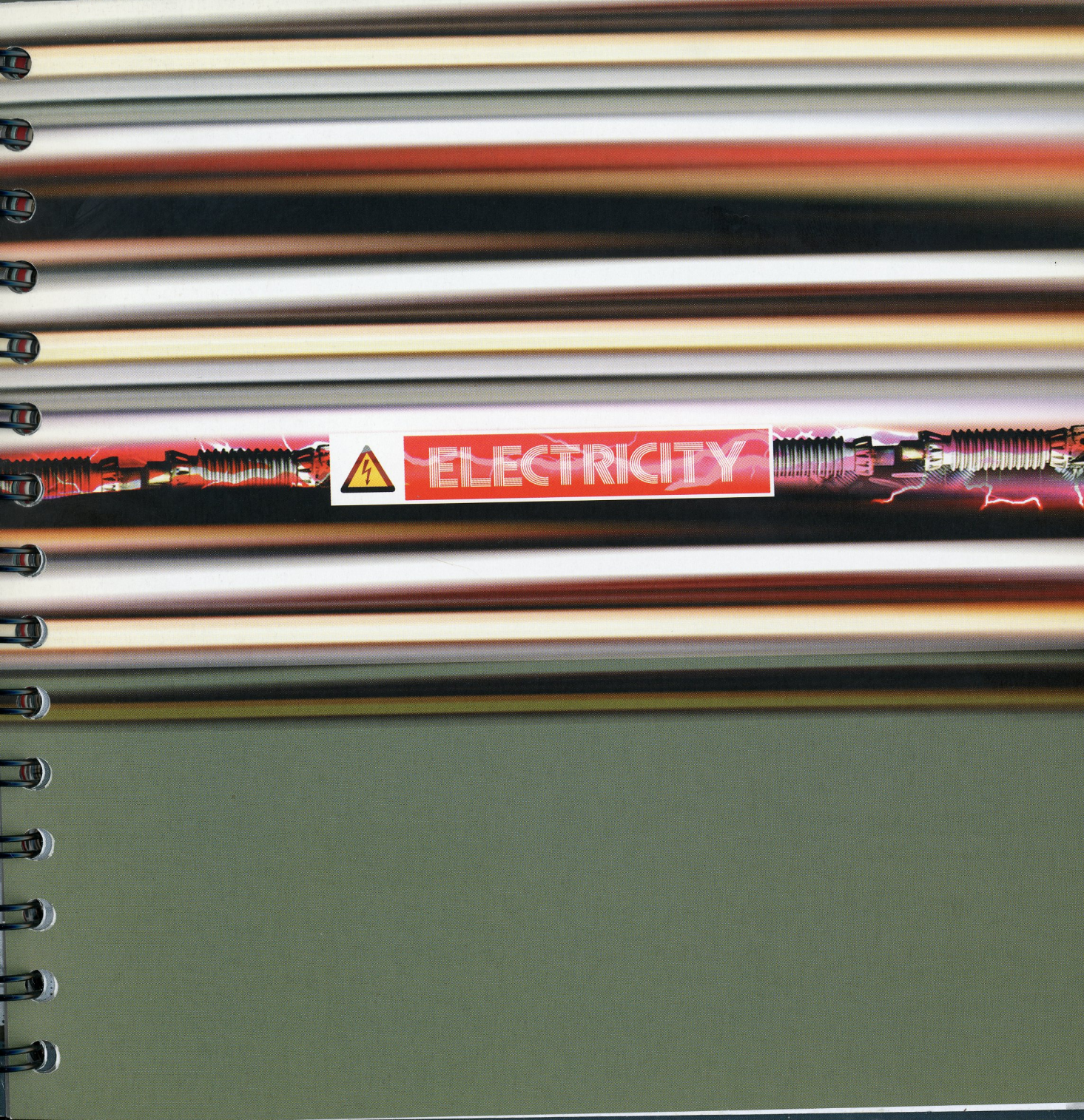
Below are a number of instructions for utilising the full power of the system:

- You can only use the Rightmouse to select IpoKeys in the IpoWindow. Border select and extend select are also enabled here. Select all IpoKeys to transform the complete animation system in the 3DWindow.
- The 'Insert Key' always affects all selected Objects. The IpoKeys for multiple Objects can also be transformed simultaneously in the 3DWindow.
- Use the SHIFT+K command: "SHOW AND SELECT ALL KEYS" to transform complete animations of a group of Objects all at once.
- Use the PAGEUP and PAGEDOWN commands to select subsequent keys in the 3DWindow.
- You can create IpoKeys with each arrangement of channels. By consciously excluding certain channels, you can force a situation in which changes to key positions in 3DWindow can only be made to the values specified by the visible channels. For example: with only the channel "LOCX" selected, the keys can only be moved in the X direction.

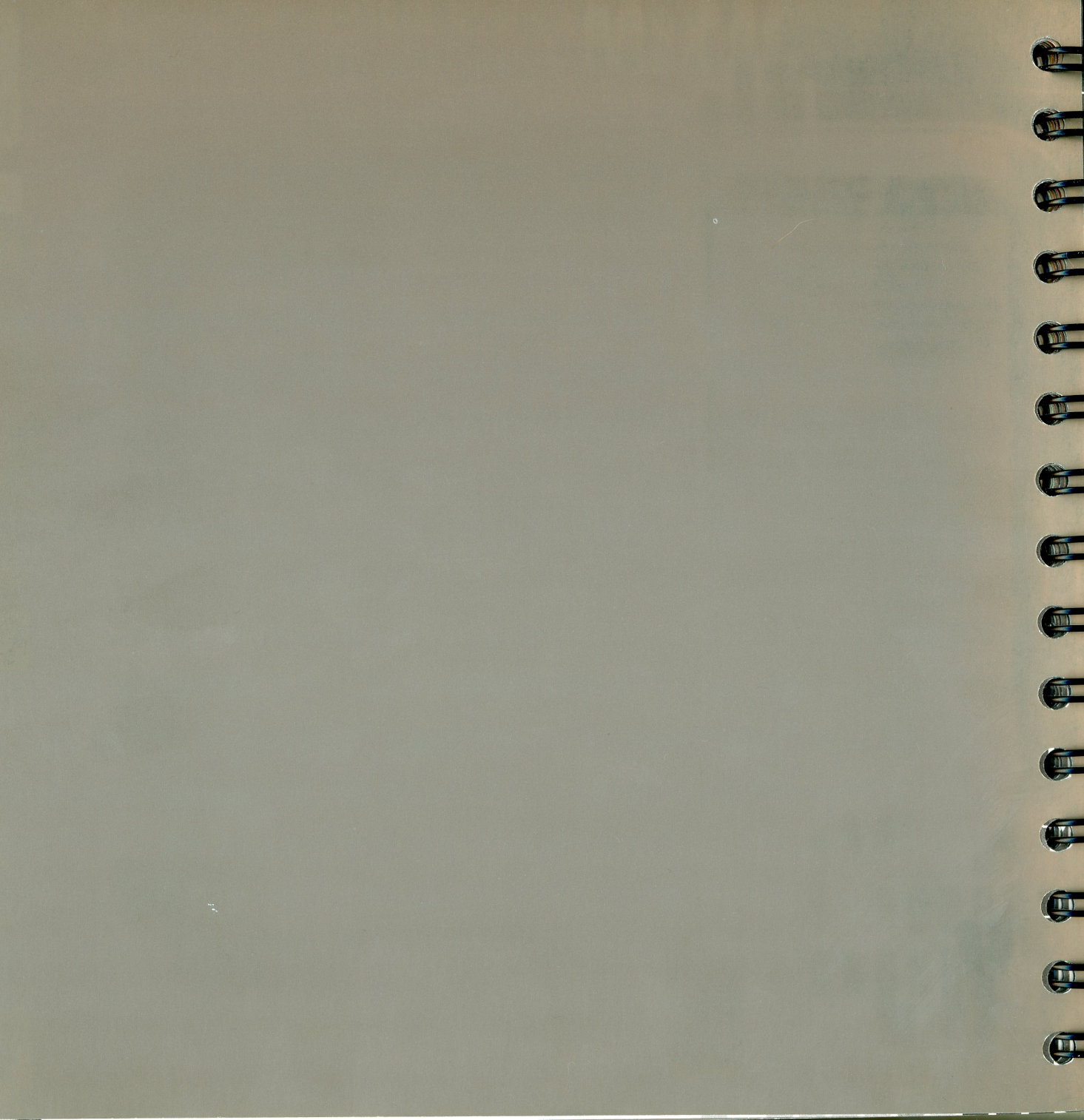
Each IpoKey consists of the vertices that have exactly the same frame value. If vertices are moved manually, this can result in large numbers of keys, each of which only have one curve. In that case, use the JKEY ["JOIN"] command to combine selected IpoKeys.

It is also possible to assign selected IpoKeys vertices for all the visible curves: use IKEY in the IpoWindow and choose "SELECTED KEYS"

The DrawKey option and the IpoKey mode can be switched on and off independently. Use the button EditButtons→DrawKey to switch off this option or Object. You can switch IpoKey mode on and off yourself with KKEY in the IpoWindow. Only KKEY in the 3DWindow turns on/off both the DrawKey and IpoKey mode.



ELECTRICITY



VERTEX KEYS

THE KEY BLOCK

VertexKeys, which should not be confused with "OBJECT KEYS" the specified positions of Objects, can also be created in Blender. VertexKeys are the specified positions of vertices in ObData. Since this can involve thousands of vertices, separate motion curves are not created for each vertex, but the traditional Key position system is used instead. A single IpoCurve is used to determine how interpolation is performed and the times at which a VertexKey can be seen.

VertexKeys are part of ObData, not of an Object. When duplicating ObData, the associated VertexKey block is also copied. It is not possible to permit multiple users to use VertexKeys in Blender, since it would not be very practical.

The Key block is also universal and understands the distinction between a Mesh, Curve or Lattice. Their interface and use are therefore identical. Working with Mesh VertexKeys is explained in detail in this section, which also contains a number of brief comments on the other ObData.

The first VertexKey position created, is always the reference Key. This key defines the texture coordinates. Only with this Key active the faces and curves or the number of vertices can be changed. It is allowed to assign other Keys a different number of vertices. The Key system automatically interpolates this.

MESH VERTEXKEYS

Creating VertexKeys in Blender is very simple, but the fact that the system is very sensitive in terms of its configuration, can cause a number of 'invisible' things to happen. The following rule must therefore be taken into consideration:

- As soon as a VertexKey position is inserted it is immediately active. All subsequent changes in the Mesh are linked to this Key position. It is therefore important that the Key position is added before editing begins.

A practical example is given below. When working with VertexKeys, it is very handy to have an IpoWindow open. Use the first Screen from the standard Blender file, for example. In the IpoWindow, we must then specify that we want to see the VertexKeys. Do this using the Icon button with the vertex square. Go to the 3DWindow with the mouse cursor and press IKEY. With a Mesh Object active, this key gives us the "INSERT KEY" menu with the "MESH" option at the bottom. As soon as this has been selected, a yellow horizontal line is drawn in the IpoWindow. This is the first key and thus the reference Key. An IpoCurve is also created.

Go a few frames further and again select: IKEY, ENTER (in the 3DWindow). The second Key is drawn as a light blue line. This is a normal Key; this key and all subsequent Keys affect only the vertex information. Press TAB for EditMode and translate one of the vertices in the Mesh. Then browse a few frames back: nothing happens! As long as we are in EditMode, other VertexKeys are not applied. What you see in EditMode is always the active VertexKey.

Leave EditMode and browse through the frames again. We now see the effect of the VertexKey system.

VertexKeys can only be selected in the IpoWindow. We always do this out of EditMode: the 'contents' of the VertexKey are now temporarily displayed in the Mesh. We can edit the specified Key by starting Editmode.

There are three methods for working with Vertex Keys:

1. The 'performance animation' method.
This method works entirely in EditMode, chronologically from position to position.
Insert Key. The reference is specified. A few frames further: Insert Key. Edit the Mesh for the second position. A few frames further: Insert Key. Edit the Mesh for the third position. Etc.
2. The 'editing' method.
We first insert all of the required Keys, unless we have already created the Keys using the method described above. Blender is not in EditMode.
Select a Key. Now start EditMode, change the Mesh and leave EditMode. Select a Key. Start EditMode, change the Mesh and leave EditMode. Etc.
3. The 'insert' method
Whether or not there are already Keys and whether or not we are in EditMode does not matter in this method.
Go to the frame in which the new Key must be inserted.
Insert Key.
Go to a new frame, Insert Key.
Etc.

While in EditMode, the Keys cannot be switched. If the user attempts to do so, a menu appears: "COPY KEY" This method can be used to copy the current key to the newly selected Key.

THE IPOCURVE AND VERTEXKEY LINES

Both the IpoCurve and the VertexKey lines are drawn in the IpoWindow. They can be separately selected with RightMouse. Since it would otherwise be too difficult working with them, selection of the Key lines are switched off when the curve is in EditMode.

The channel button can be used to temporarily hide the curve (SHIFT+LeftMouse on "SPEED") to make it easier to select Keys.

The Key lines in the IpoWindow can be placed at any vertical position. Select the line and use Grab mode to do this. The IpoCurve can also be processed here in the same way as described in the previous section. Instead of a 'value' however the curve determines the interpolation between the Keys, e.g. a sine curve can be used to create a cyclical animation.

TIPS

- Key positions are always added with IKEY, even if they are located at the same vertical position. Use this to copy positions when inserting. Two key lines at the same position can also be used to change the effect of the interpolation.

- If no Keys are selected, EditMode can be invoked as usual. However, when you leave EditMode, all changes are undone. Insert the Key in EditMode in this case.

- For Keys, there is no difference between selected and active. It is therefore not possible to select multiple Keys.

- When working with Keys with differing numbers of vertices, the faces can become an enormous chaos. There are no tools that can be used to specify precise sequence of vertices. This option is actually suitable only for Meshes that have only vertices such as Halos.

- EditButtons'Slurph is an extremely interesting option. The "Slurph" number indicates the interpolation of Keys per vertex with a fixed delay. The first vertex comes first and the last vertex has a delay of

"Slurph" frames. This effect makes it possible to create very interesting and lively Key framing.

Pay special attention to the sequence of the vertices for Meshes. They can be sorted using the command EditButtons'Xsort or made random using the command EditButtons'Hash. This must of course be done before the VertexKeys are created. Otherwise, unpredictable things can happen (great for Halos though).

CURVE AND SURFACE KEYS

As mentioned earlier in this manual, Curve and Surface Keys work exactly the same way as Mesh Keys. For Curves, it is particularly interesting to place Curve Keys in the bevel Object. Although this animation is not displayed real-time in the 3Dwindow, it is rendered.

LATTICE KEYS

Lattice Vertex Keys can be applied in a variety of ways by the user. When combined with "SLURPING", they can achieve some very interesting effects. As soon as one Key is present in a Lattice, the buttons that are used to determine the resolution are blocked.

With a Key line selected, three interpolation types can be specified. Press TKEY to open a menu with the options:

"LINEAR" interpolation between the Keys is linear. The Key line is displayed as a dotted line.

"CARDINAL" interpolation between the Keys is fluid; the standard.

"BSPLINE" interpolation between the Keys is extra fluid and includes four Keys in the interpolation calculation. The positions are no longer displayed precisely, however. The Key line is drawn as a broken line.

SPECIAL ANIMATION TECHNIQUES

MOTION PATHS

Curve Objects can be used for the 3D display of an animation path. Only the first curve in the Object is then used. Each Curve becomes a path by setting the option AnimButtons→CurvePath to ON. All Child Objects of the Curve move along the specified path. It is also a good idea to set the option EditButtons→3D to ON so that the paths can be freely modeled. In the ADD menu under Curve→Path, a primitive with the correct settings is already. This is a 5th order Nurbs spline, which can be used to create very fluid, continuous movements.

Curves that are used as paths always get the correct number of interpolated points automatically. The button EditButtons→ResolU has not effect here.

The speed along a path is determined with a curve in the IpoWindow. To see it, the Header button with the 'arrow' icon must be pressed in.

The complete path runs in the IpoWindow between the vertical values 0.0 and 1.0. Drawing a curve between these values links the time to the position on the path. Backward and pulsing movements are possible with this.

→117

For most paths, an IpoCurve must run precisely between the Y-values 0.0 and 1.0. To achieve this, use the Number menu [NKEY] in the IpoWindow.

If the IpoCurve is deleted, the value of AnimButtons→PathLen determines the duration of the path. A linear movement is defined in this case.

Using the option AnimButtons→CurveFollow, a rotation is also given to the Child Objects of the path, so that they permanently point in the direction of the path. Use the "TRACKING" buttons in the AnimButtons to specify the effect of the rotation:

TrackX Y Z -X -Y -Z UpX Y Z

TRACKX, Y Z, -X, -Y -Z (RowBut)

This specifies the direction axis, i.e. the axis that is placed on the path.

UPX, UPY UPZ (RowBut)

Specifies which axis must point 'upwards' in the direction of the [local] positive Z axis. If the "TRACK" is the "UP" axis, it is deactivated.

Curve paths cannot be given uniform rotations that are perpendicular to the local Z axis. That would make it impossible to determine the 'up' axis.

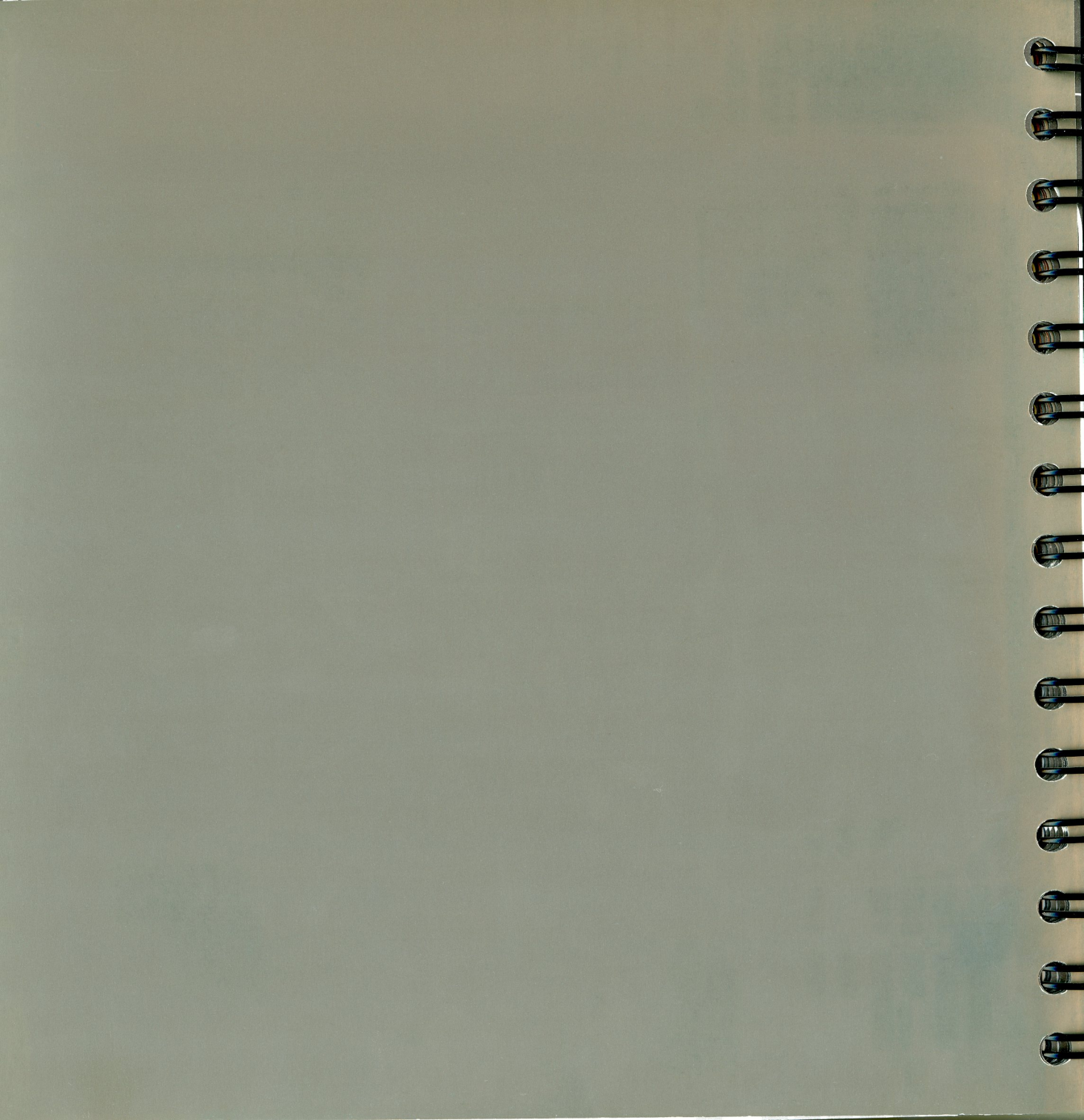
To visualise these rotations precisely, we must make it possible for a Child to have its own rotations. Erase the Child's rotation with ALT+R. Also erase the "PARENT INVERSE" ALT+P.

The best method is to 'parent' an unrotated Child to the path with the command SHIFT-CTRL+P: "MAKE PARENT WITHOUT INVERSE" Now the Child jumps directly to the path and the Child points in the right direction.

3D paths also get an extra value for each vertex: the 'tilt' This can be used to specify an axis rotation. Use TKEY in EditMode to change the tilt of selected vertices in EditMode, e.g. to have a Child move around as if it were on a rollercoaster

BLENDERED,
NOT STIRRED!





OBJECT TRACKING

In Blender Objects can be given a rotation constraint so that they always refer to a 'Track' Object.

Activate the option in the 3DWindow with CTRL+T. "MAKE TRACK" All selected Objects now refer to the active Object.

Since Objects can also have their own rotation, this rotation can better be erased first: ALT+A. If the Object is a Child, erase the "PARENT INVERSE" also:

ALT+P.

Then use the tracking buttons again to specify the desired tracking effect.

The tracking constraint is the 'odd man out' in the hierarchy. Parent rotations

and the rotation of the Object itself can affect tracking. The option EditButtons→PowerTrack can be used to ignore the Object and Parent rotations.

Tracking is often used with Cameras to make sure that the main subject is always visible. The disadvantage of tracking is that it occurs perfectly. The object being followed always appears in the centre of the picture, making it appear to stand almost completely still. Use of manually inserted Object lpos in Cameras gives a more natural effect. Carefully chosen camera positions and subtle camera movements can have a significant impact on the power of an animation.

DUPLICATORS

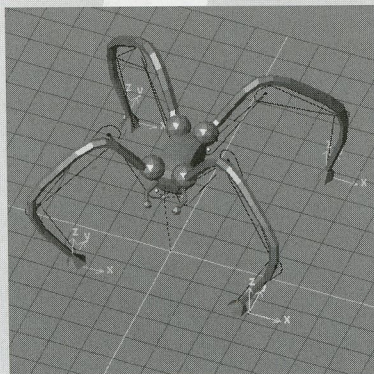
Blender can automatically generate Objects without actually duplicating them. This places a 'virtual' copy of the Object on each specified frame of an animation system. It is also possible to have a virtual copy placed on each vertex [or particle]. These options can be specified in the AnimButtons.

The first option is called "DUPLIFRAMES". Whatever type of movement the Object has, i.e. with its own Object lpos or along a Curve path, a temporary copy of the Object is created for each frame, from "DUPSTA" to "DUPEND". During the specified frame interval, the "DUPLIFRAMES" system is created. The automatically generated Objects are displayed in a light grey wireframe.

The second option is "DUPLIVERTS". This option works only for Mesh Objects. A copy of the Child Object is placed on each vertex of the Mesh. The option also works for Meshes that emit Particles. A copy is placed on each Particle.

The command CTRL-SHIFT+A ["MAKE DUPLI'S REAL"] makes the generated Objects real. This tool is very useful when modeling.

IKA AND SKELETONS



The methods that can be used to build stable Ikas and Skeletons were handled in the previous section.

→106

Ikas can be put in motion with hierarchies. By parenting the 'Effectors' to another Object, hand movements and foot steps can be defined quite easily. In the illustration above, Effectors on the legs have been parented to the Emptys.

An alternative is to 'key frame' the Effector itself. Three channels are available in the standard Object Ipo for this: EffX, EffY, EffZ. The IKEY ["INSERT KEY"] menu also offers this option.

The way in which the Ika is currently implemented makes it possible to effectively control individual movements and timing, but it is very important that the global animation system is clear in advance in terms of precisely where it is located and how the rotation will be.

PARTICLES

Particles are halos (or Objects if the option "DUPLIVERTS" is ON) that are generated based on the laws of nature. Particles can be used to create smoke, fire, explosions, a fountain, fireworks or a school of fish.

A Particle system is precalculated as a pre-process (it can sometimes take a while), after which it can be viewed in the 3Dwindow in real-time. Particles are drawn as small black pixels.

Blender does not separately calculate and remember the locations of all of the particles in each frame for the particle system. Instead, interpolation occurs between a fixed number of key positions. A larger number of keys gives a more fluid, detailed movement, but also significantly increases the amount of memory used and the time required to calculate the system.

Particles are a full-fledged part of Blender's animation system. They can also be deformed and controlled by Lattices.

Only Meshes can have Particles.

Use the AnimButtons to add an "EFFECT" of the type "PARTICLE". Then specify the desired settings:

The total number of Particles to be generated. The first Particle is created on the position of the first vertex, etc.. The process is performed cyclically, vertex by vertex.

The start time of the first Particle and the total time that Particles are generated.

The age of each Particle, which can be varied by a random factor

The start speed of the Particle, based on the vertex normal, a random factor the current Object speed or even a Texture.

The force that affects the Particle for the duration of its life. The force may be gravity or a sort of wind. Textures can contribute a lively turbulence.

The way in which the Particle is drawn, i.e. as a 'point' or as a line rounded in the direction in which the Particle moves.

→270

Next, a Halo Material must be added to the Mesh. Any of the possible halos can

be Particles. See also the description of the Halo MaterialButtons.

Material lpos can be used to completely change the colour intensity or size of Particles during their life span. By default, the lpo time interval of 0-100 visualises the entire life of a Particle.

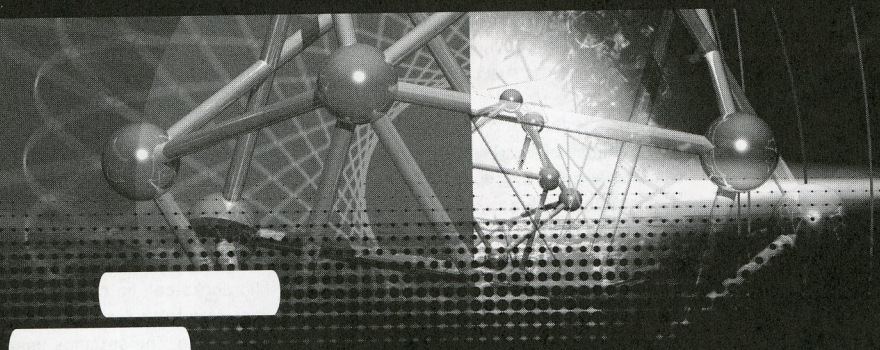
Blender does not offer presets for 'smoke' 'fire' or 'explosions'. The Particle system is flexible and interactive enough to master the fine points with a little practice.

A number of tips are given below:

The start speed can be determined by the vertex normals or the Mesh. If the

Mesh does not have faces (and thus has no vertex normals), the normalised local vertex coordinate is used as start speed.

- Use a negative "Z FORCE" to suggest gravity. Add a "DAMP" factor to this as friction.
- Fire and smoke Particles are very convincing if they are given a twisting movement. This can be achieved by applying a Texture to the movement:
- By default, this is the last Texture of the Material, in channel number 8. Use a procedural Texture such as "CLOUDS" or "MARBLE".
- The Particle Button "TEX" determines the strength of the Texture's effect.
- Set the Particle Button "GRAD" to ON (the gradient). This is a mathematical derivative based on 4 samples that together form a speed vector. This method gives a very convincing turbulent effect with procedural textures.
- The button "NABLA" determines the size of the area in which the gradient is calculated. This value must be carefully tuned to the frequency of the texture.
- Set the number of "KEYS" as high as possible to view the extremely subtle twists this method sometimes produces
- The turbulence can be displayed more clearly in the 3DWindow if the Particle option "VEST" is selected. Note, however, that this also has an impact on the rendering method.
- Standard halos are always rendered as a 'sum total'. By giving MaterialButtons→Add a lower value, dark halos can be rendered.
- Smoke halos can be created most effectively using an Image Texture of a cloud of smoke. Set the "ADD" and "ALPHA" value of the Material to the lowest possible value. Clouds of smoke start out small and gradually increase in size and become more transparent as they age.
- The best explosions can be achieved by superimposing different Particle systems with differing speeds and colours.
- Fireworks can be created by allowing Particles to 'multiply' at the ends of their lives. The settings used to achieve this are located to the right of the "CURMUL" button.
- The option AnimButtons→DupliVerts places a 'virtual' copy of the Child Objects of the Particle Mesh on each Particle. Setting the "VEST" option to ON also gives a rotation.



- 1. Introduction
- 2. Materials
- 3. Lighting
- 4. Rendering

Part 6

Materials, light: render!



MATERIALS AND TEXTURES

RENDERING

Rendering consists of three steps:

1. Blender determines what is visible for each pixel, e.g. what faces or halos.
2. The colour is calculated for all visible faces and halos.
3. The various [colour] layers are combined.

This section examines the second of the three steps, which is the core of the rendering process: the colour calculation, which is also called 'lighting' or 'shading'

SHADING

In general terms, we can say that as soon as light shines on something, that thing becomes visible. What we see is actually the light that is reflected. Complex calculations have been developed to create this effect. Blender uses a special mix of formulas to achieve an acceptably convincing impression of reality.

The 'shading' calculation includes:

- Ambient light.
The simplest form of light, a constant factor.
- Directional light
Light with constant direction, e.g. sunlight.
- Positional light
Light with a position in the space, a point light source or a spot beam.
- Light source attenuation
The *distance* to the lamp determines the intensity.
- Diffuse reflection.
The intensity is determined based on the *angle* at which the light shines on a face. The calculation compares the normal vector of the face to the lamp vector.
- Specular reflection
This is the shine, actually a sort of

'mirroring' of the light source in the object that it shines upon. For the first time, the view point plays a role here. The shine is at its maximum if the reflection from the view point in the face points towards the lamp.

- Normal interpolation
Also called 'Phong' shading; the vertex normals within a face are interpolated to achieve fluid *shading*. The vertex normals are determined by calculating the average of the plane normals that surround them.
- Gouraud shading
The colours of the vertices are interpolated within a face. In this case, a complex light calculation may be performed in advance for each vertex, e.g. using a radiosity model.

In Blender, the variables required for lighting are indicated in the following DataBlocks:

- Material
All colour, specularity, reflection and transparency data van faces and halos. A Material determines the way in which a Texture is controlled and processed.
- Lamp
The colour and intensity of the light, special types of light beams and shadow. Lamps can also have Textures.

- World

The 'ambient' light, a global exposure time, mist and standard background skies. Textures can also be used here.

- Texture

The universal block that can be used to control variables from a Material, Lamp or World.

MATERIALS

The settings of the MaterialsButtons are described in the reference section of this manual. A number of settings for the creation of standard material are provided below as an example:

→238

[Images on next page]

Plastic has a high reflection and sharp shine. Set the "REF" "SPEC" and the "HARD" to high.

Paper hardly shines at all. Since it reflects light very uniformly, the "EMIT" value can be used.

Metal typically has a relatively low *reflectivity* and has a soft, wide shine with a colour that differs slightly from the colour of the metal itself.

Blender can be used to create glass without RayTracing. The glass effect becomes visible when a very low "ALPHA" value is combined with a high "SPTR" value (the glass becomes opaque where shine is present).

Add a reflection map as a finishing touch.
Too bad that we cannot actually make
glass refraction!

The "Halo" option can be used to create a wide variety of different light effects, including lens flares.

Combined with the Particles Effect, a Material can also be used to create **smoke**, water driplets, fireworks or explosions. In this example an Image Texture is used as a cloud of smoke. A very low "ALPHA" and "ADD" allow us to create a very realistic smoke effect with a few dozen of Particles.

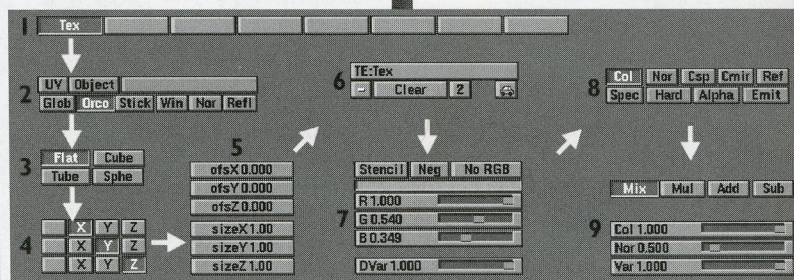
THE 'MAPPING'

effect of the Texture on the Material is specified.

The MaterialButtons on the right-hand side (and the Lamp and World buttons) are reserved for the *mapping*. The buttons are organised in the sequence in which the 'texture pipeline' is performed.

1. Texture channels

Each Material has eight *channels* to which Textures can be linked. Each *channel* has its own individual *mapping*. By default, textures are executed one after another and then superimposed. A second Texture *channel* can completely replace the first one.



In Blender the Materials and Textures form separate blocks. This approach was selected to keep the interface simple and to allow universal integration between Textures, Lamps and World blocks. The relationship between a Material and a Texture is called the 'mapping'. This relationship is two-sided. First, the information that is passed on to the Texture must be specified. Then the

plastic

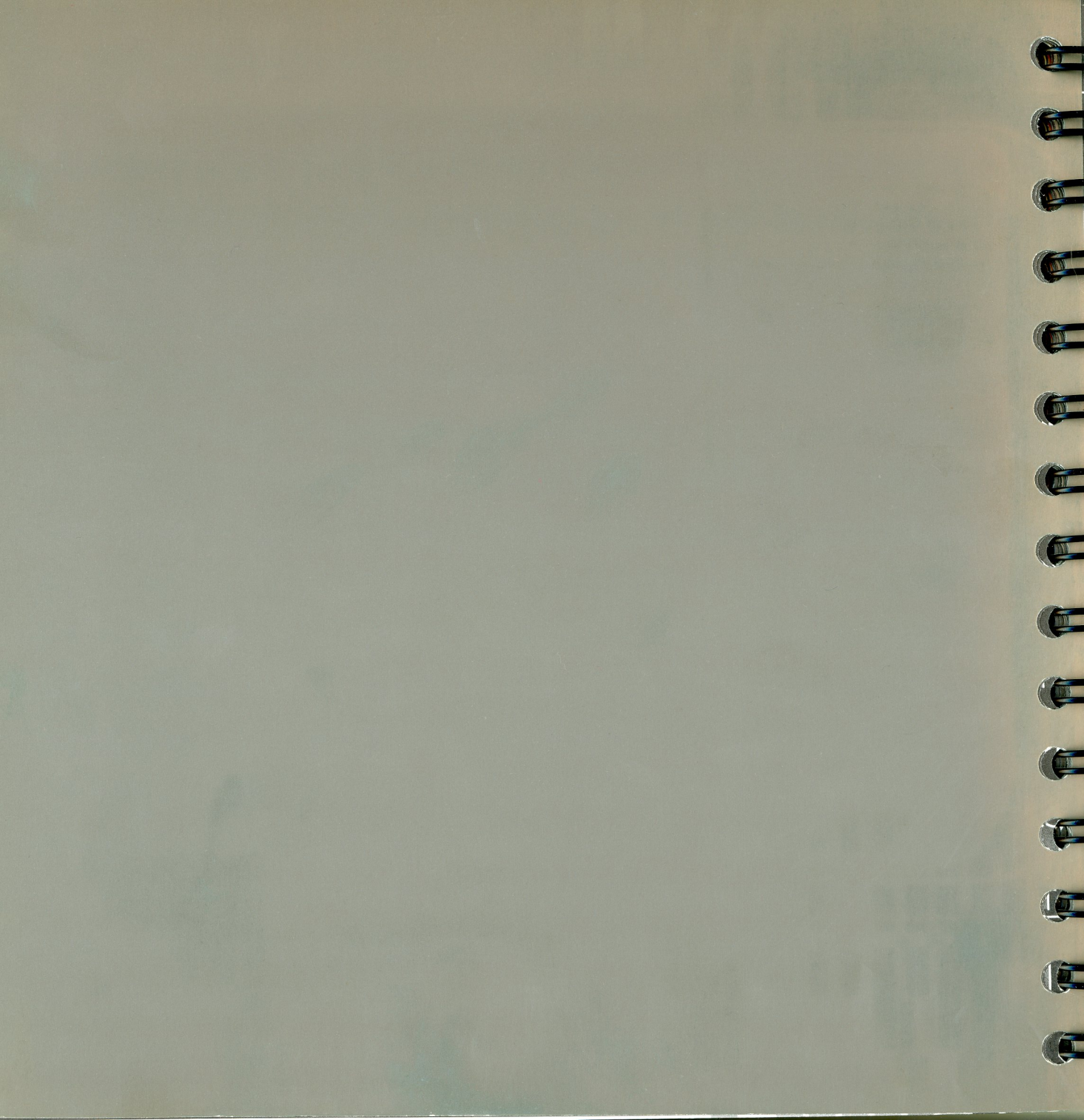
paper

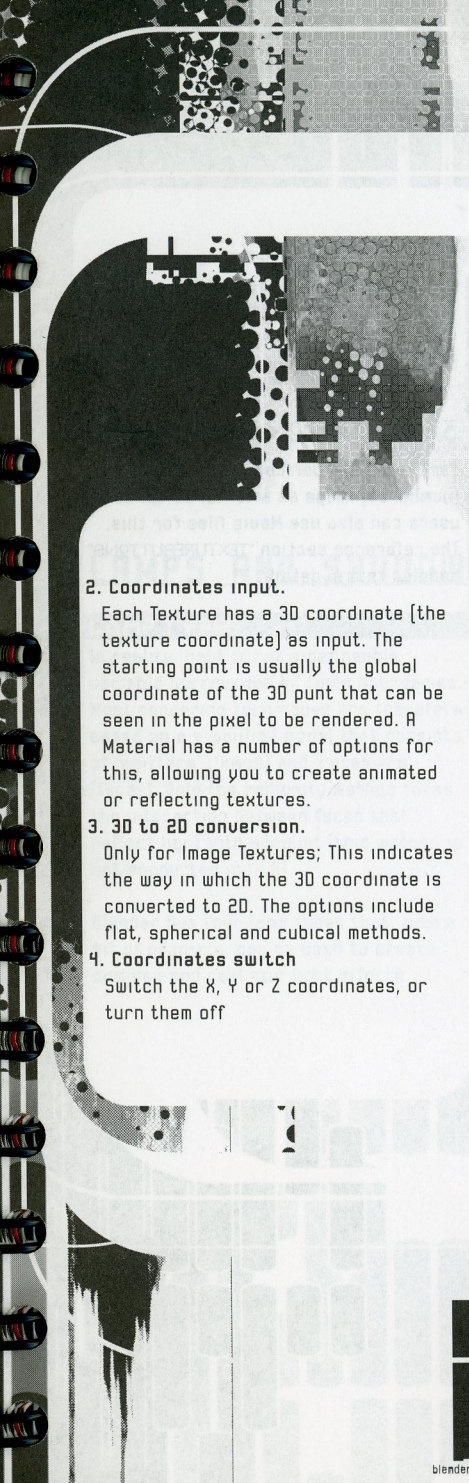
metal

glas

halo

smoke





2. Coordinates input.

Each Texture has a 3D coordinate [the texture coordinate] as input. The starting point is usually the global coordinate of the 3D point that can be seen in the pixel to be rendered. A Material has a number of options for this, allowing you to create animated or reflecting textures.

3. 3D to 2D conversion.

Only for Image Textures; This indicates the way in which the 3D coordinate is converted to 2D. The options include flat, spherical and cubical methods.

4. Coordinates switch

Switch the X, Y or Z coordinates, or turn them off

5. Coordinates transform.

The texture coordinate can be given an extra translation or scaling.

6. The Texture itself

Only the name of the Texture block must be specified.

7. Texture input settings.

A number of standard settings for the Texture, plus specification of the effect of the current Texture on the next one.

8. Mapping: output to.

All prior data are used to change parts of the Material. Textures can therefore have an effect not only on colour, but on the normal (bump mapping) or the "Alpha" value as well.

9. Output settings

Indicates the strength of the effect of the Texture output. Mixing is possible with a standard value, addition, subtraction or multiplication.

TEXTURES

Textures give three types of output:

- Intensity textures: return a single value. This intensity can control "ALPHA", for example, or determine the strength of a colour specified using the *mapping* buttons.
- RGB textures: return three values, they always affect colour.
- Bump textures: return three values, they always affect the normal vector. Only the "STUCCO" and "IMAGE" texture can give normals.

A distinction is also made between 2D textures and 3D [procedural] textures. "WOOD", for example, is a *procedural* texture. This means that each 3D coordinate can be translated directly into a colour or a value. These types of textures are 'real' 3D, they fit together perfectly at the edges and continue to look like what they are meant to look like even when cut; as if a block of wood has really been cut in two. Procedural textures are *not* filtered extra or anti-aliased. This is hardly ever a problem: the user can easily keep the specified frequencies within acceptable limits.

ANIMATED MATERIALS

The Image texture is the only 2D texture and is the most frequently used and most advanced of Blender's textures. The standard, built-in bump mapping and perspective-corrected MipMapping, filtering and anti-aliasing guarantee outstanding images (set DisplayButtons→OSA to ON for this). Because pictures are two-dimensional, the way in which the 3D texture coordinate is translated to 2D must be specified in the *mapping* buttons.

There are three ways to create an animated Material:

1. MATERIAL IPOs

Just as with Objects, IpoCurves can be used to specify 'key positions' for Materials. With the mouse in the ButtonsWindow, the command IKEY calls up a PopMenu with options for the various Material variables.
→117

Per Material, the mapping for all 8 channels can be controlled with IpoCurves. A continuously rising curve can be placed in the channel "OFFSZ" for example, to create a water-like effect in an X-Y face.

2. OBJECTS THAT GIVE TEXTURE COORDINATES

Each Object in Blender can be used as a source for texture coordinates. To do this, the option "OBJECT" must be selected in the green "COORDINATES input" buttons and the name of the Object must be filled in.

An inverse transformation is now performed on the global render coordinate to obtain the *local* Object coordinate. This links the texture to the location, size and rotation of the Object.

3. ANIMATION IMAGES

Per frame, Blender can load another [numbered] Image as a texture map. SGI users can also use Movie files for this. The reference section "TEXTUREBUTTONS" handles this in detail.

Lamp Spot Sun Hemi

LAMPS AND SHADOW

LAMP TYPES

In reality, light is not a defineable variable surrounded by fixed boundaries. Most rendering techniques are therefore based on a simplified model that consists of 'emitters' (lamps) and 'receivers' (faces). Only the radiosity method takes the interaction between faces that reflect light into account (this method is not supported in V1.5).

Blender has four lamp types that, with a bit of practice, can be used to create complex and realistic light effects.

LAMP

The standard lamp, a pointed light source that shines uniformly in all directions without throwing shadows. The variable 'Dist' determines the distance in which the Lamp shines.

A large number of carefully hung and slightly coloured lamps can give an environment a warm 'radiosity' effect.

SPOT

The light is limited to a cone-shaped area. The 3DWindow shows the form of the beam with a dotted line. Use the sliders "SPOTSZ" and "SPOTBL" to modify the angle and intensity of the beam. This is the only type of lamp that can throw shadows. This limitation is caused by the calculation method. See the following section for additional information.

SUN

The light shines from a constant direction; distance has no effect. The position of the Lamp Object is therefore unimportant. Only the rotation is important.

HEMI

Similar to "SUN", but now the light shines in the form of a half-circle, a *hemisphere*. This method is also called *directional ambient*. It can be used to suggest cloudy daylight.

See the description of the LampButtons for a complete overview of all lamp options.

→233

SHADOW BUFFERS

Blender uses the well-known *shadow buffer* algorithm for shadow calculations. A picture is rendered from the spot lamp; only the distances from the lamp are saved for each pixel. By comparing the location of the pixel to be rendered with the same location seen from the lamp, the system determines whether the pixel receives light or is in the 'shadow'. The shadow buffers save a 24 bit number for each distance, compressed in groups of 8*8 pixels. A buffer of 1024*1024 pixels therefore only requires an average of 1.5 Mb memory.

A shadow lamp is drawn in the 3DWindow with a 'clipping line'. This line indicates the area, seen from the lamp, in which the shadow calculations occur. Everything in *front* of the line, with the value "CLIPSTA" *always* has light. The faces *farther* than the end point of the line, with the the value "CLIPEND", *always* have shadow.

Due in part to the fact that the resolution of the lamp buffer is limited, these values must be carefully chosen. A high value for "CLIPSTA" has a positive effect in term of quality.

The shadow buffer method works very quickly, but it has two possible side effects:

- *Aliasing*. The shadow edge has a block-like progression. This can be solved by making the spot beam smaller or by increasing the buffer resolution or the number of *samples* in the buffer.
- *Biasing*. Faces that are in full light display *banding* in a block-like pattern. Set the "BIAS" value as high as possible and decrease the distance between "CLIPSTA" and "CLIPEND".

Shadow lamps can be efficiently adjusted by having them temporarily function as Camera in a 3DWindow. Use the command CTRL+NUMPAD.0 on the *active* Lamp to do this. The 3DWindow now shows interactively the clipping boundaries and the spot beam of the lamp.

The command ALT+NUMPAD.0 restores the previous Camera Object.

The variables are described in detail in the section LampButtons. Remember to set the global option DisplayButtons→Shadow to ON.

SELECTIVE LIGHTING

By default, all of the lamps in the visible layers throw light on all Objects. The option LampButtons→Lay limits this. With this option, only Objects in the same layer[s] as the Lamp Object receive light. This makes it possible to light objects selectively to give them an extra accent or to limit the effect of lamps to a specific space. This option also helps to keep render times under control.

LAMP TEXTURES

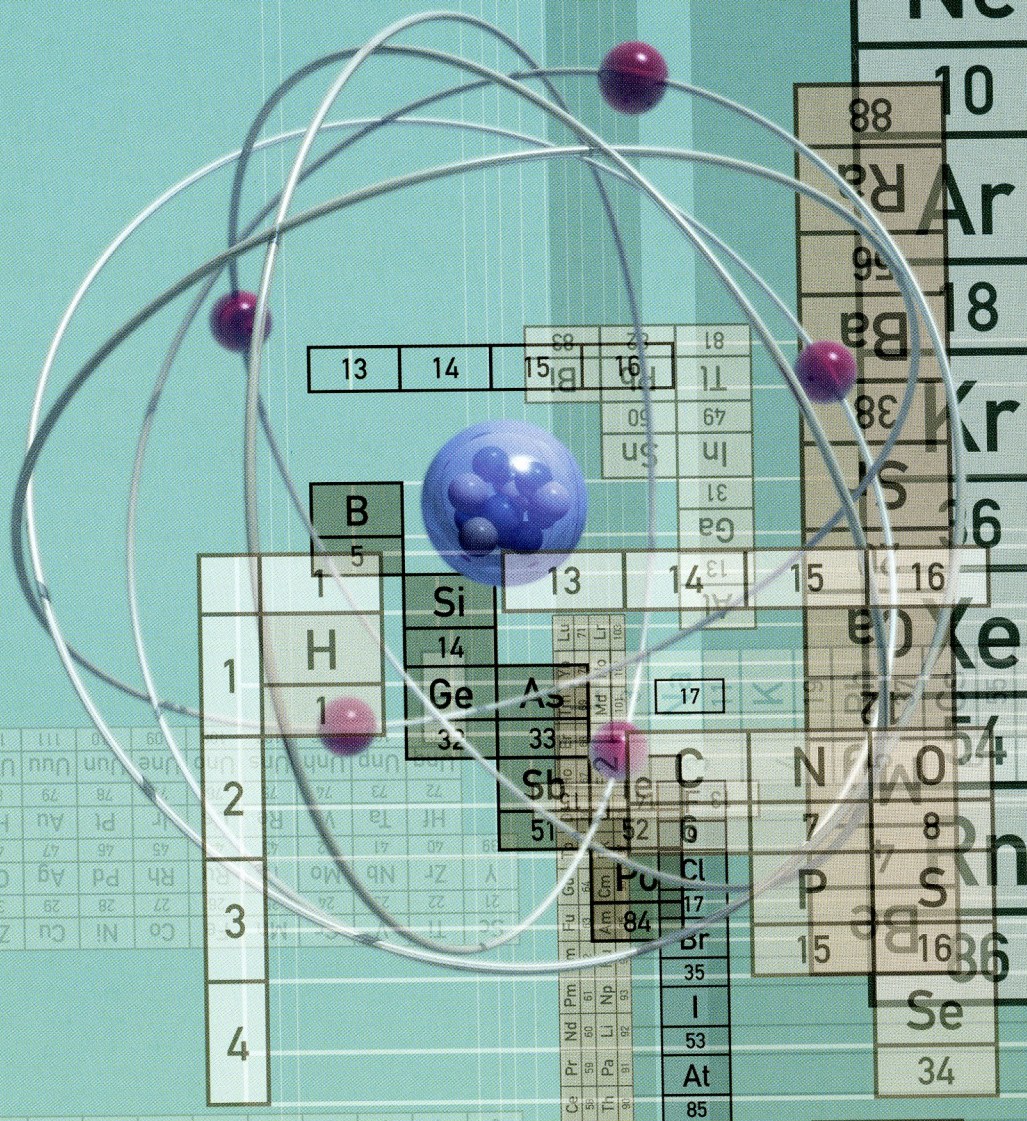
Each Lamp has six *channels* to which Textures can be linked. Each *channel* has its own *mapping*; the effect of the texture on the lamp. The number of settings is limited in comparison with those of a Material. The only "MAPPING OUTPUT" for example, is the lamp colour. The structure and effect of the *mapping* is otherwise identical to the *mapping* for Materials described earlier in this section.

The creation of a somewhat 'stained' light is usually enough to allow you to create advanced light effects. The spot Lamp gives a 'correct' texture coordinate as an extra, which means that it can function as a slide projector: select the "COORDINATES INPUT" option "VIEW". This can be used to create 'underwater' effects or in combination with an Image Texture, to create a 'leaded glass' effect.

Per Lamp, the *mapping* for all 6 channels can be controlled with IpoCurves. A continuously rising curve can be placed in the channel "OFFSZ" for example, to create a moving underwater effect with a Clouds Texture.

LIGHTING TIPS

Use the CTRL+Z command in the 3DWindow to force a 'Gouraud shaded' picture. This often results in enough feedback to adjust the lamps. If necessary, Meshes can be subdivided a number of times to achieve acceptable resolution.



He

2

Ne

10

88

Ar

95

8

Kr

10

6



erke

7

540

8
§ Rn

22

1686



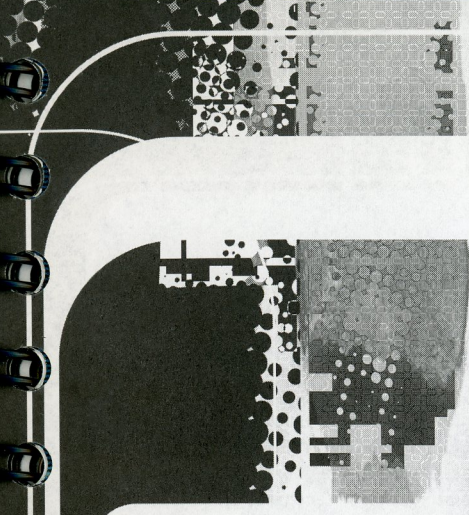
Se



34

2





The variables in Blender are adjusted in such a way that at least two lamps are required to achieve sufficient contrast.

Use coloured light to suggest depth. Blue light looks like it is farther away than warm light. A single blue light placed diagonally *behind* an object can work wonders.

Use of *ambient* light (in the WorldButtons) should be limited to specifying a colour temperature. Make everything cooler by selecting a dark blue ambient, or warmer by selecting a dark red ambient.

An alternative for using ambient is using a Hemi Lamp. This prevents

environment light from shining too uniformly from all directions.

- Each lamp has its own "Energy" value. Large increases in this value for certain lamps can increase the contrast in your rendering and make it more dramatic.
- Use the option WorldButtons→Exposure to give the entire rendering more or less contrast. To prevent confusion, however, use this option when making final adjustments.
- Shadow is absolutely essential to give objects 'body': a clear position and visible relationship with the environment. Use of one out of three lamps as a shadow lamp is sufficient in many cases.
- If the total light situation is already acceptable, extra shadow lamps can be placed with the option "ONLYSHADOW".
- A Sun that throws shadow can be suggested by placing a very small spot beam at a great distance. Remember to take the "ClipSta" and "CLIPEND" variables into account.
- The Spot lamp is the only lamp for which a halo beam can be rendered. The halo beam is based on the circumference of the lamp beam and is calculated to fit into the 3D

environment properly. A spot beam does shine 'through' faces, however, since it does not make use of a shadow algorithm. The range of the spot halo is determined based on the value of "Dist".

- Lamps have Ipos that can be used to animate all relevant settings. With the mouse in the ButtonsWindow, the command IKEY calls up a PupMenu with options for the various variables.

RENDERING AND POST-PRODUCTION

VIDEO WORK

Blender has a peculiar mix of render processes that are designed for optimum speed and manageability. This type of design reveals Blender's past as an in-house tool for a video animation studio. Only the rendering methods that were required and that could be realised within the limited time period posed by the customer were realised. The use of all kinds of *fancy* techniques for glass and mirroring (ray-tracing) or complex 'solid' 3D textures for rust, fur and hair look great in demos and stills, but simply did not fit into the budgets of NeoGeo's customers.

The main advantage of the current render engine, and all of Blender's features in fact, is that it was designed specifically for day-to-day 3D work. Blender is a tool designed to *work* with, to design quickly and to visualise in 3D easily. Blender was designed to fulfil the changing requirements of the customer and designer efficiently and effectively. I must admit, however that the current render engine is somewhat 'antique'. It is one of the few pieces of C-code that was taken out of the predecessor of Blender with minor changes. We have been planning to drastically modernise the render engine for the last two years, but have not been able to find the time!

I hope that releasing texture plug-ins in the upcoming Blender version will appease enthusiastic Blender users. The next logical step is to completely release the sources and/or drastically modernise the renderer with a library API.

This section examines the entire render process. I felt that it would not only be useful for users to know how it works, but that it would also be a good idea to finally clarify this somewhat over-mystified process.

FACES AND NORMALS

Whatever their type, all 3D Objects are transformed into triangular and rectangular faces for rendering. These faces, together with the Halos, are the primitives that are used to construct the rendering.

The render faces are always two-sided in Blender. It does not matter what position the face normal is actually in, it is always displayed as if the normal is pointing in the direction of the camera. Both the inside and outside of a hollow tube can be correctly displayed using this approach.

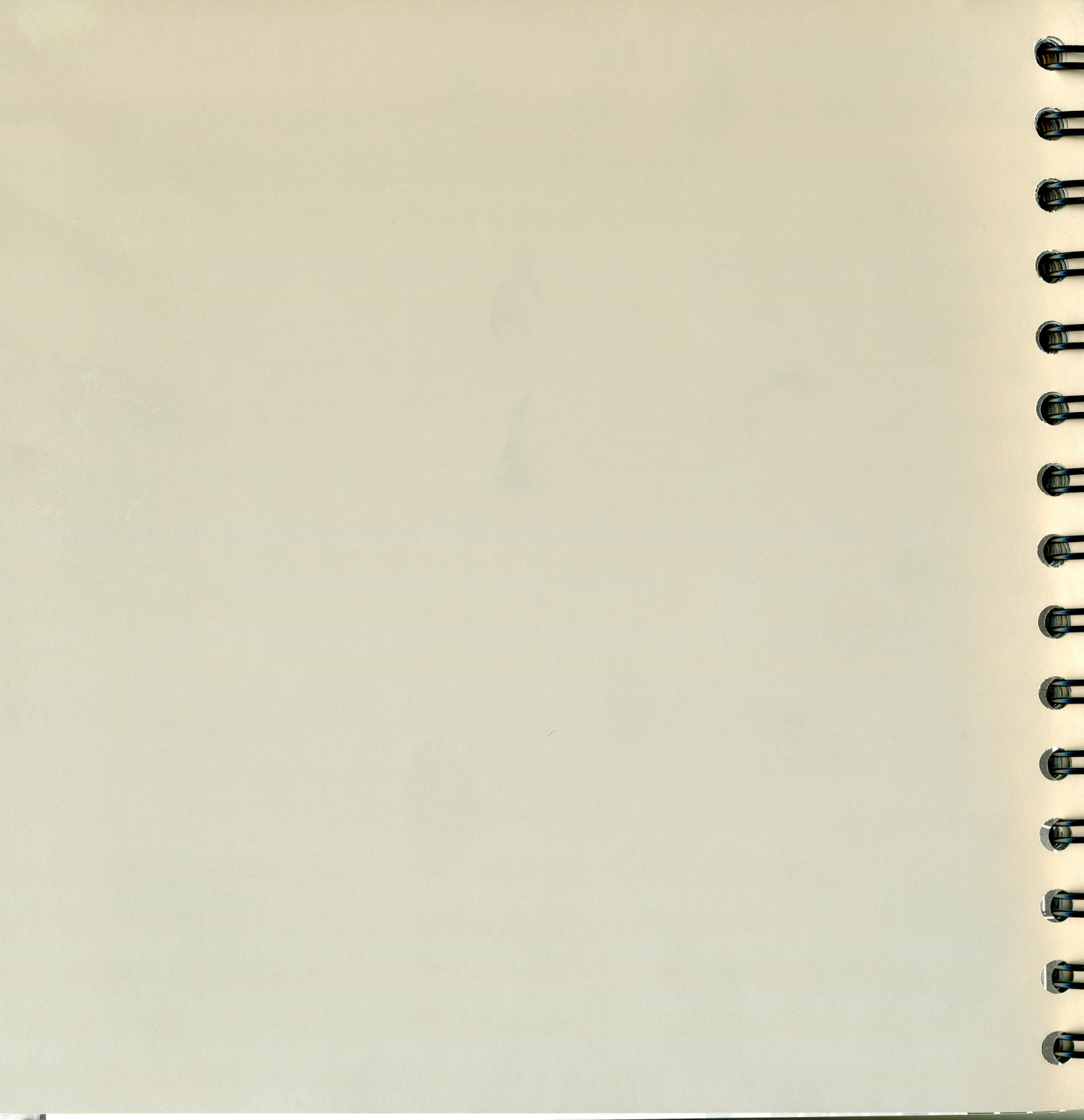
To prevent individual faces from being visible in fluidly curved surfaces, the vertex normals in faces can be interpolated; the *smoothing*. Vertex normals of procedural Objects such as Surfaces and MetaBalls are always correctly calculated. The vertex normals of a Mesh, however are derived according to the geometry. An irregular distribution of faces may therefore cause minor discontinuities in the *smoothing*.

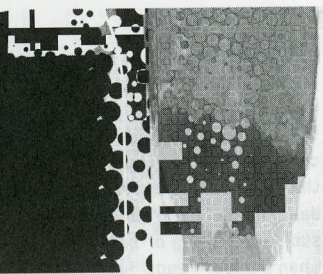
Each time you leave EditMode, the vertex normals in a Mesh are calculated. This may cause problems if the Mesh has a large number of faces with sharp angles, or with normals that randomly face both sides. The correct direction of the vertex normals can no longer be determined in this situation.

For these Meshes, it is important that the face normals all point the same way: in EditMode, use CTRL-N. The button "NOFLIP" can then be used to specify that the Mesh must be considered 'single-sided' when the vertex normals are calculated.



particled





THE RENDER ENGINE

As soon as the "RENDER" process is started in Blender the following procedures are performed (the situation is described below based on the assumption of maximum render quality):

Sequences.

If the option "DO SEQUENCE" has been selected, the strips from the Sequence Window are loaded, evaluated and executed. In other cases, the main render loop is started.

Outermost main loop: Fields and Motion Blur.

Each video field and each Motion Blur step is a complete rendering, each with required time difference.

- Main loop: Lamps, faces and halos.

All Objects from the Scene are evaluated; the animation system is set to the correct time and the hierarchies and deformations are fully calculated. The ultimate result is a quantity of render faces, render halos and render lamps with *global* coordinates.

- Main loop: Shadow buffers.

All shadow buffers that are present are created using the global coordinates of the faces, making it insensitive to the *camera view* transformation.

- Main loop: Parts and view calculation

The picture to be rendered is then divided into the desired parts. This offers opportunities to save memory ("PARTS") and to create panoramas. A separate 3D, perspective and viewport transformation is calculated for each 'part'.

- Render loop: coordinates.

All render faces, lamps and halos are transformed.

- Render loop: ZbufferDA.

For each "OSA" sample, a complete Zbuffer and an associated buffer with colour codes for the faces are created. Only the differences between the buffers are remembered by the system [delta accumulation].

- Scanline loop: ZbufferDA-Alpha.

The transparent faces are processed in blocks of 64 scanlines with a modified ZbufferDA routine. The routine sorts all transparent faces by 'Z' and compares the results with the normal Zbuffer. All transparent faces are completely rendered by pixel and added with *alpha*.

- Scanline loop: 'solid' faces.

Per scanline; all opaque faces are rendered and combined with the scanline transparent faces.

- Scanline loop: Halos.

Per scanline; all halos are compared to the normal Zbuffer, rendered and added with the faces. This is an important point: halos always lie *over* transparent faces!

- Scanline loop: Sky.

Per scanline: the background 'sky' or "BACKGROUND" is inserted.

- Main loop: Parts.

The parts or panoramas are added.

- Main loop: Flares.

The lens flares are superimposed over the picture.

- Outermost Main loop.

The fields and Motion Blur layers are added.

WITHIN A PIXEL

The pixels of a scanline in the render process can contain multiple faces and colours. These can be transparent layers or they may be required for anti-aliasing. The combination of all of these colours occurs in 64 bit resolution with gamma correction. The gamma correction guarantees a 'pure' colour depth and 'shimmer-less' anti-aliasing with a standard 24 bit video card, without the need for dramatic [post-]corrections.

The following calculations are performed for each pixel:

- For each face:
 - The exact 3D coordinate is calculated.
 - The vertex normals are interpolated.
 - The texture coordinates are calculated.
 - All Texture channels are processed. Now a test is performed to determine whether Image files must be loaded.
- The texture plug-ins also play a role here.
- Texture calculations result in a completely modified 'render' Material block.
- For each Lamp:
 - The lamp textures are calculated
 - The shadow buffer is evaluated.
 - All shading calculations with the 'render' Material are performed.

This results in the 4 colour components.

- The lamp halos are added.
 - The mist factor is applied to the alpha.
- All colours of the faces are weighted and gamma corrected added.

ALPHA

The result of a rendering is always a 32 bit RGBA picture. In the *alpha* layer the amount of 'coverage' of the pixel is specified in 256 gradations. This is important not only for transparent faces, but particularly for correct display and storage of anti-aliasing information.

The *alpha* layer can only be written in graphics if the option "RGBA" is ON. *Alpha* must be used to allow the various rendered layers to be superimposed over one another as post-process.

Prior to rendering, the way in which the alpha must be delivered must be specified. The DisplayButtons offer three options:

"SKY"

If there is a World with 'sky' it is filled in in the background. The *alpha* is not changed, but the transparent colours 'pollute' the colour of the background, making the picture less suitable for post-processing.

"PREMUL"

'Sky' is never filled in and the *alpha* in the picture is delivered as "Premul" - a white pixel with an *alpha* value of 0.5 is then: [RGBA bytes] 128, 128, 128, 128. The colour values are thus multiplied with the *alpha* value in advance.

Use "PREMUL alpha for post-processing such as filtering or *scaling*. When Blender reads RGBA, "Premul" is used as the standard.

"KEY"

'Sky' is never filled in and the *alpha* and colour values remain unchanged. A white pixel with an *alpha* value of 0.5 is then: [RGBA bytes] 255, 255, 255, 128. What this means is particularly clear when rendering Halos: all transparency information is in the [invisible] *alpha* layer. Many drawing programs work better with "KEY" alpha.

Because halos normally *add* colour it is not actually possible to effectively express this with *alpha*. The *alpha* value determines only the coverage level, the mix value of a colour. For this reason, in post-process added halos (and halos in

'sky') look different from directly rendered halos.

The option "XALPHA" is included in the MaterialButtons for the creation of a more extreme alpha progression for post-process halos.

When Blender is used for 'compositing' the combination of [video] graphics in layers with alpha masks, the internal gamma-corrected rendering has a negative effect on the alpha. In such cases, the option DisplayButtons→Gamma can be set to OFF.

MEMORY

During and after rendering, the memory used is displayed in the InfoHeader. This information is an excellent way to determine whether problems may occur (or have already occurred). Blender is unable to handle its own memory limit. If insufficient memory is available, the program shuts down with the message "MALLOC RETURNS NULL". In such cases, start by making sure that sufficient swap space is available and make sure that the Unix limit is not too low [set it as follows: "LIMIT MEMORYUSE UNLIMITED"].

Then check the major drains on memory:

- Render resolution: 10-16 bytes per pixel.
- Shadow buffers: 1024*1024 costs approximately 1.5 Mb.
- Image Textures: minimum 4 bytes/pixel.
- Faces: rule of thumb is 100 bytes per face, including vertices.

A common error is to underestimate the enormous number of faces that can result from Beveling; simple Curves can generate 20k of faces in no time. The same is true of Text Objects, Nurbs and MetaBalls. Decreasing the number of faces not only saves memory, it also has a positive effect on rendering time.

The built-in render limits (V1.5):

- Render faces: 1024k
- Render vertices: 1024k
- Render halos: 1024k
- Render lamps: 256

These limits apply only to the amounts that can be *rendered*. Within Blender itself (the Blender file), the only limitation is the available computer memory.

GRAPHICS FILE FORMATS

Blender supports the following graphics file formats:

- HamX

A self-developed 'lossy' 8 bits RLE format. The files are very compact and can be displayed very quickly. Primarily for use with the "PLAY" option.

- Targa

The most commonly used standard. This format, which is RLE compressed, can be used for 8 bits (grey values), 24 bits RGB and 32 bits RGBA.

- Iris

The SGI software standard. This format, which is also RLE compressed, can be used for 8 bits, 24 bits RGB and 32 bits RGBA.

- JPEG

Only for 24 bits graphics. This *lossy* format offers high compression. Use the "QUALITY" button to specify the desired compression value.

- Movie

Only for the SGI version: Movie files, a sequential series of JPEG images.

Use an "FTYPE" file to indicate that this file is an example of the type of graphics format you want Blender to write pictures to. This makes it possible to process 'colormap' formats; the colormap data are read from the file and used to convert the 24 or 32 bit graphics. If the option "RGBA" is set, colour number '0' is used as transparent colour by default.

Use Ftypes voor:

[Amiga] IFF The entire range: the 2-8 bits colormaps and the HAM format.

Targa Colormap

[SGI] Iris ColorMap

CDI RLE colormap and DYUV format.

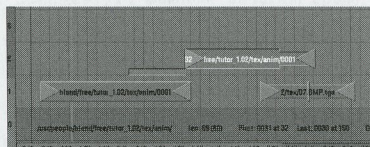
SEQUENCE EDITOR

Each Scene in Blender offers the option of executing special effects on pictures from it or even creating a complete montage; the Sequence editor

Each Sequence (strip) is a random set of pictures that can be moved in time and which each have an independent start and end point. Various strips are combined with Effect strips. A number of presets are available for this, including "CROSS" and "ALPHAOVER"

A Scene renders either directly from the 3D data or it executes Sequences. This is determined by the option

DisplayButtons→"DO SEQUENCE"



The standard SequenceWindow has a grid with time on the horizontal axis and the layers (or 'machines' in video terms) on the vertical axis. Layer 0, the lowest layer displays text containing information on the selected *strip*.

Strips can be manipulated in Blender using the standard commands that are available through Blender: RightMouse to select, GKEY to translate, SHIFT+A to add a

new strip, SHIFT+D to create a copy and XKEY to delete strips.

A Sequence can be lengthened or shortened using the triangular 'sliders' on the ends of the Sequence. They can be separately selected.



To view the result of the Sequences, a second Sequence window can be created with the Header button "DISPLAYIMAGE" ON. The third Screen in the standard Blender startup file is pre-configured for Sequence editing.

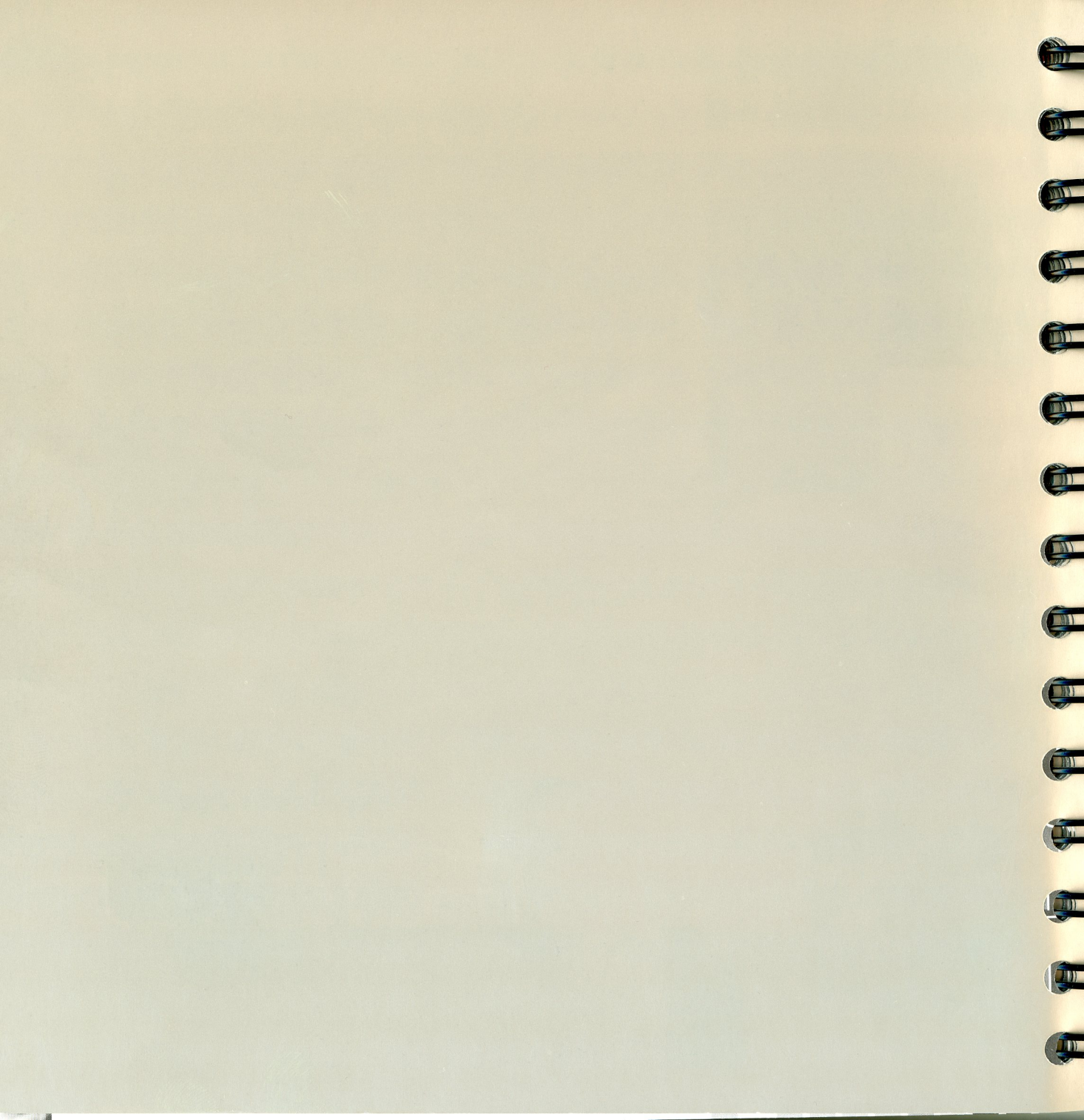
As long as a picture has not been rendered (with F12), all strips that have been loaded or calculated remain in the memory, allowing you to view 'previews' in real-time.

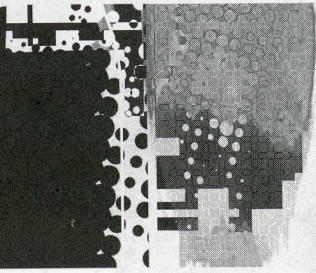
The contents of all of the strips, with the exception of the one on the current frame, are released only if a frame is rendered, provided the option DisplayButtons→"DO SEQUENCE" has been selected.



EVENT 2958. 1278.
X 09676

התחלה
של





An Effect can be added by selecting two overlapping strips and indicating the type of effect using SHIFT+A:

"CROSS": a fluid progression from strip 1 to strip 2.

"GAMMACROSS": this is a gamma-corrected cross. It gives a more 'natural' progression in which lighter sections are inserted before darker ones.

"ADD": two strips are added.

"SUB": the second strip is subtracted from the first one.

"MUL": the strips are multiplied.

"ALPHAOVER": the second strip with its *alpha* is superimposed over the first one. Generally, pictures with *alpha* are 32 bits.

"ALPHAUNDER": the first strip is inserted after the second one, with the *alpha* of the second strip.

- "ALPHAOVERDROP": same as "ALPHAOVER", but with a drop shadow.

The sequence in which an Effect uses the strips can be changed with the CKEY. By default, the Effect is linear. An lpo can be linked to the Effect to specify any type of progression you require.

→117

Groups of selected strips can be combined to a Meta strip. Use the command MKEY to do this. This can be done only if no *unselected* strips are linked to the selection with effects.

Use TABKEY to view the contents of a Meta and to subsequently leave the Meta.

Metas can appear in other Metas and behave exactly like a normal Sequence strip. When Metas are duplicated, their contents are not linked.

Use the command ALT+M to undo the Meta.

Multiple strips or effects can be present in each frame, but each one must be at a different height. The calculation order when displaying a frame is:

- first all Images or Movies are loaded on *the current frame* [if this has not already been done].
- then all Effect strips are calculated on *the current frame* [if this has not already been done].
- the *uppermost* Effect strip is always displayed.
- If there is no Effect strip: the *lowest* Sequence strip is displayed. This makes it possible to save 'temporary' Sequences at a higher position without having them influence the picture.

The contents of a strip can be displayed on the current frame by clicking the strip with LeftMouse and keeping the mouse button pressed in.

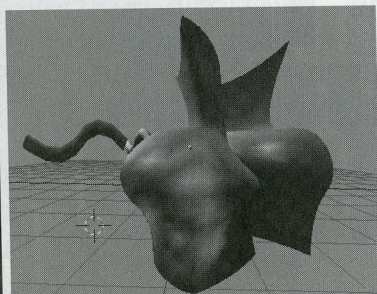
The result of Sequences is always displayed in the resolution set in the DisplayButtons with "SIZEX" and "SIZEY". If the option DisplayButtons→OSA is ON, the the pictures are filtered enlarged or filtered reduced.



Part 7

Special techniques

VERTEX PAINT



The VertexPaint option makes it possible to assign individual colours to each vertex interactively, as if you were painting a Mesh. This is displayed in the rendering using Gouraud interpolation.

Vertex colours can replace the Material colour or be added to the shading calculation as 'light'. The latter option is particularly interesting for radiosity effects. Use the MaterialButtons to specify the desired method.

To ensure correct use of the VertexPaint option, the Mesh should be set to 'Single sided' [EditButtons]. All normals in the Mesh should also point in the same direction (use CTRL+N in EditMode).

Use the button EditButtons → "MAKE VERTCOL" to specify that vertex colours must be used. From that moment onwards, the vertex colours are remembered by the application, even in EditMode. If the Mesh was already displayed as 'Shaded' [CTRL+Z in the 3DWindow], the existing colours are copied to the vertex colours. This makes it possible to save 'light' in the vertices and to use VertexPaint to draw shadows without needing to use Lamps.

UKEY can be used to start and stop VertexPaint mode. In this mode, the 3DWindow works like a drawing program:

LeftMouse (hold, draw):
to draw the active colour.

MiddleMouse:
is unchanged; for view rotation or translation.

RightMouse (click):
sample a colour, this is the active colour.

UKEY:
Undo, this is a 'real' undo; UKEY restores the situation when used a second time.

KKEY:
Clear, all vertices are set in the drawing colour.

Use the VertexPaintButtons to specify the active colour and settings for the drawing operation. They are described in detail in the reference section.

Contrary to what one might expect based on their name, the vertex colours are saved *per face*. This makes it possible to use different colours on faces that share vertices. This sometimes results in unexpectedly 'sharp' delineation edges. Use WKEY ["SHARED VERTEX COL"] to have all faces that share vertices blend their colours.

ROTOSCOPING

Rotoscoping is a technique that makes it possible to use 'live video' material as the basis for [character] animation. The term is based in the traditional 2D cartoon animation. 'Drawing over' video material from actors can result in very realistic animations.

In a 3D animation package, rotoscoping is a very attractive alternative for the advanced but very expensive 3D *motion capture* technique.

Rotoscoping works as follows in Blender:

- Select the required video fragment. The frames must be saved in a directory as separate, numbered files [the SGI version also supports use of Movie files].
- Create a 'help' Object with a Material and an Image Texture.
- In the Image Texture, specify:
 - The first video frame: "LOAD IMAGE" button. [for the Movie file, also activate the green "MOVIE" button].
 - The total number of frames: "FRAMES" button.
- Go to the 3Dwindow with the mouse and issue the command SHIFT+F7. The 3DWindow changes to a ButtonsWindow with settings for a "BACKGROUNDPIC".

Set "BACKGROUNDPIC" to ON and use the MenuBut at the bottom of the window to select the Image Texture.

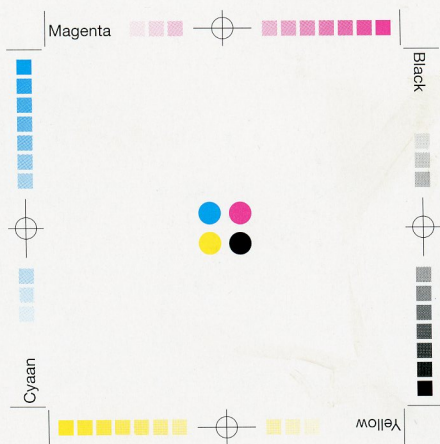
Issue the command SHIFT+F5 to restore the 3DWindow.

If the 3DWindow is in 'ortho' or 'camera' mode, the selected video frame is drawn in the background.

Rotoscoping works best from the Camera. Select a position in which the 3D model corresponds precisely with the background video.

The ArrowKeys can be used to browse through the frames. Use IKEY to animate the 3D model from position to position.

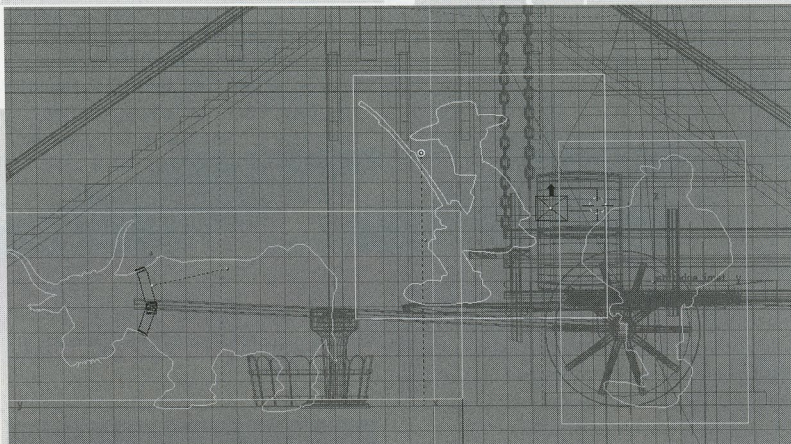
In addition to using Rotoscoping for character animation, it can also be used various *special effects*, e.g. to integrate 3D models in video or to create synchronised mouth movements.





CARTOON ANIMATION A N D 3 D

Blender was used to create two very successful animation films. The two projects involved integrating traditional cartoon animation in a 3D world. The traditional method, in which the 'background artists' supply material over which the character animations are superimposed, was turned completely upside down. The animated cartoons were placed in the 3D scene as flat 3D elements. Camera positions, movements and light were applicable as usual. The 3D animator now functions as a director determining the image cut-outs and the



moment at which the action starts and stops.

This method is also suitable for the use of *bluescreen* video material or actors.

It works as follows:

- Use the standard Mesh Plane for the cartoons. Add a Material and Image Texture to it.
- Set the Material to "SHADELESS" and "ZTRANSP". Use the green "MAPPING" buttons to specify that the Texture must affect "ALPHA". Due to the fact that the alpha comes from the graphic, the Material "ALPHA" must be set to 0.0.
- The cartoon animations can be (Amiga) DPaint anim5 files. This drawing

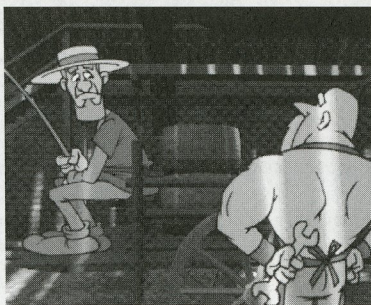
program is still very popular with animators and remains unparalleled for this type of use. In the Image Texture, activate the option "MOVIE" to indicate that the file is an anim5 file.

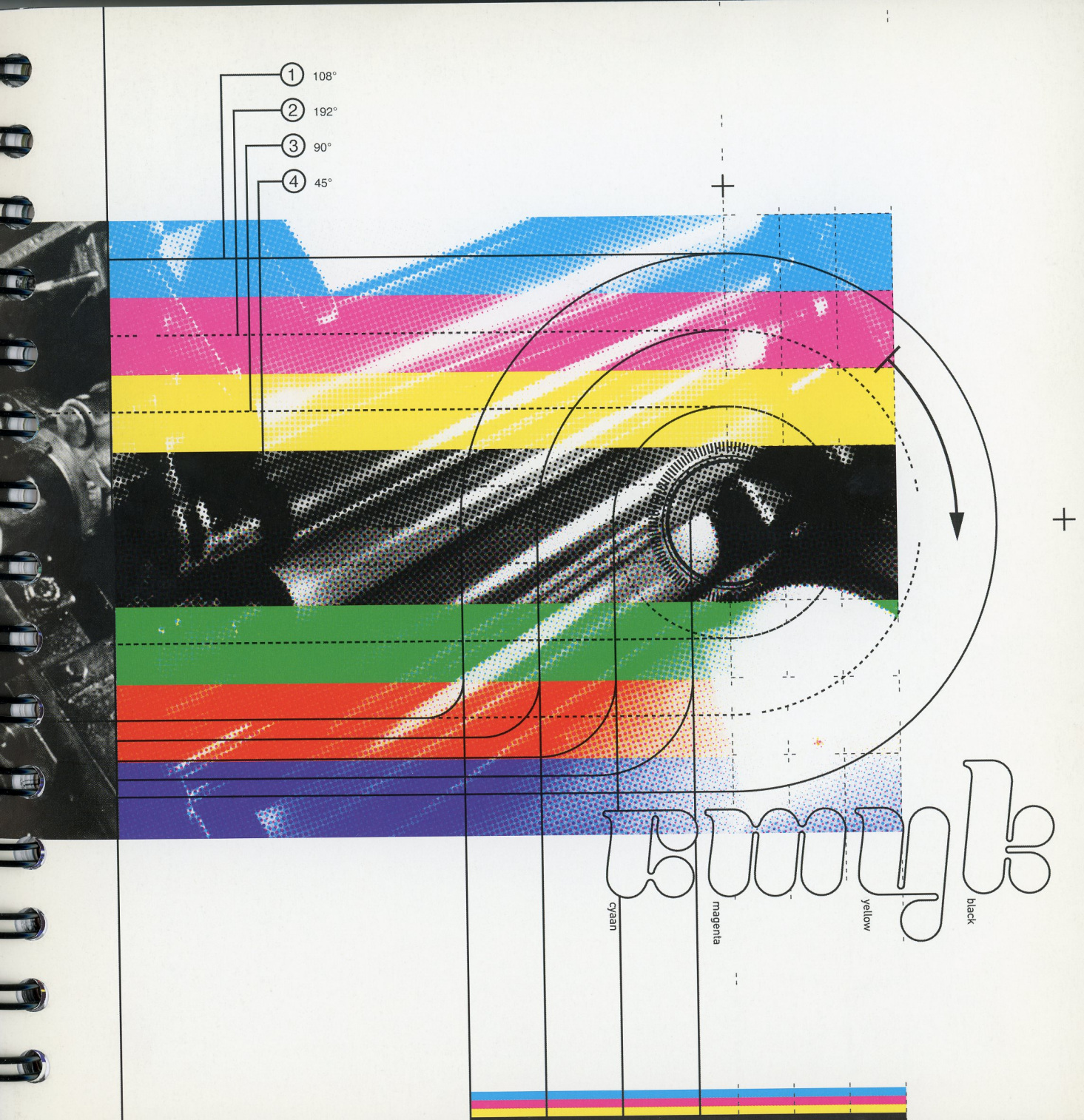
- Other types of images can also be used, provided they are also colormap files or 32 bits graphics with an *alpha* channel.
- Colour number '0' is completely transparent in colormaps.
- Colormap graphics are excellent for pre-process anti-aliasing. Select the option "ANTI" in the TextureButtons for this.
- Specify in the TextureButtons that the *alpha* must be used: "USEALPHA".

- Specify the number of frames in the TextureButtons and, if necessary, specify a playing speed and start frame.
- In the 3DWindow, issue the command ALT+V. This sets the Plane exactly to the relative dimensions of the graphics.
- Issue the command CTRL+D. The current image is now displayed in the Plane as an outline (wireframe). Use the ArrowKeys to change the current image. Each time you issue the command CTRL+D, the wireframe is modified. This makes it very easy to assign the cartoon graphic the correct position, rotation and size.

When rendering human figures, it is very important that their feet touch the floor. Since the images are flat, their feet (or shadow) must be placed slightly *below* the floor. Use a MaterialButtons→Zoffs value to mislead the Zbuffer. The graphic is given an artificial advantage over the other faces.

The "BLACKSMITH" demo file on the Blender ftp site gives a complete demonstration of this method.





① 108°

② 192°

③ 90°

④ 45°

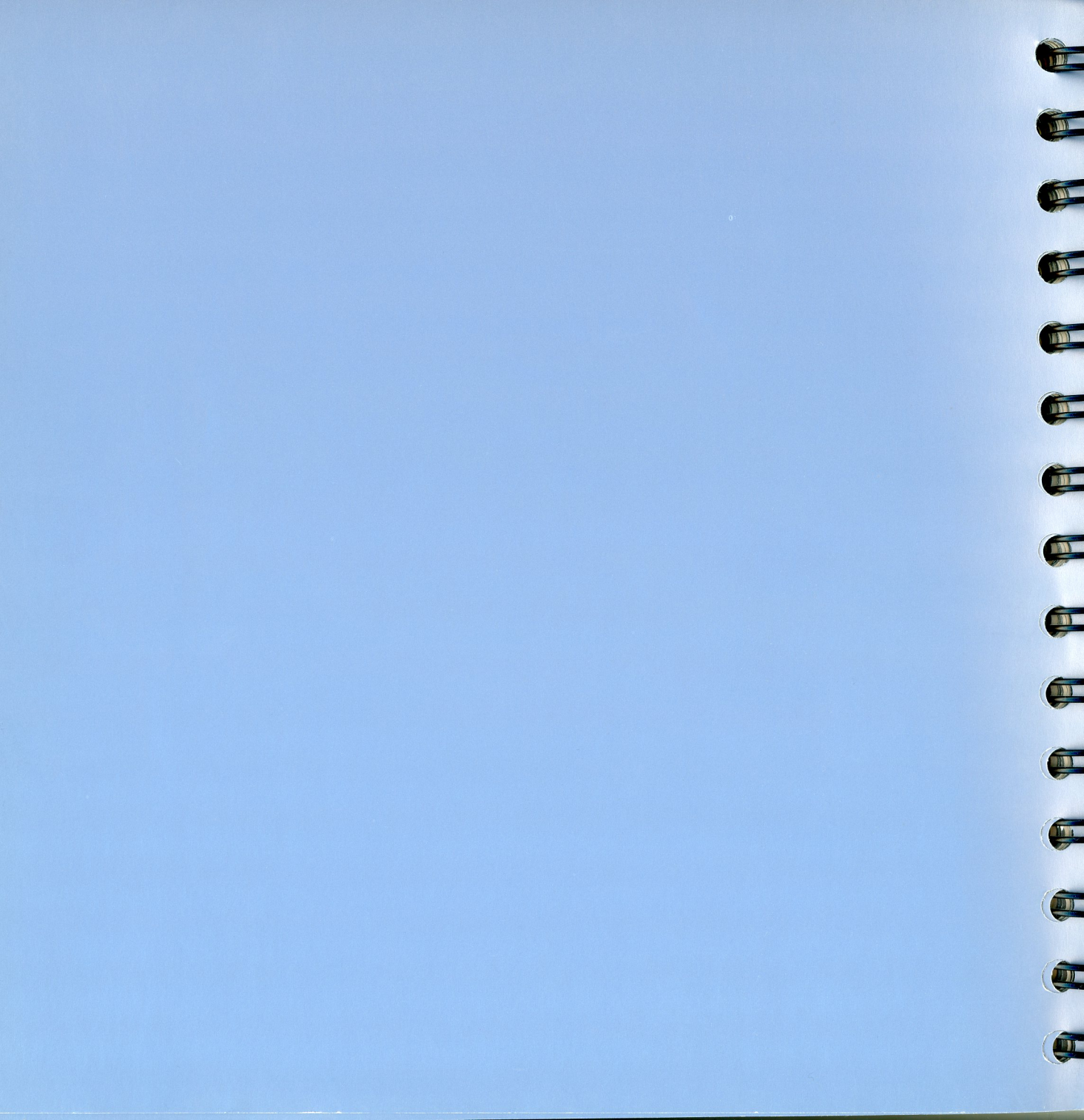
wmyk

cyan

magenta

yellow

black



I M P O R T I N V E N T O R / V R M L F I L E S

Importing 3D objects from other programs are an important feature in any 3D package. Unfortunately, the SGI past of Blender poses limitations in this area: Blender can only read Inventor files and then only Inventor 1.0 ASCII. This is a negligible limitation on an SGI, since all of the major 3D packages are supplied with a converter e.g. DxfTolV or AliasTolV. The 'ivdowndgrade' utility is also available to convert new Inventor files to 1.0. The VRML format, which is derived from Inventor and is almost identical to it, is

becoming a commonly used standard. VRML files are recognised by Blender and loaded as Inventor 1.0.

Blender automatically recognises these files, eliminating the need for the user to do anything but select the required files using F1. Each Inventor file is loaded and converted to a Mesh and/or Surface Object.

The Inventor function in Blender does not make use of a 'standard' library. The function can read the following primitives:

- FaceSet
- TriangleStripSet
- IndexedFaceSet
- IndexedTriangleMesh
- IndexedTriangleStripSet
- QuadMesh
- NurbsSurface
- IndexedNurbsSurface

The vertex colours of the Mesh types are also read. The remaining colour information is retrieved from the nodes "MATERIAL", "BASECOLOR" and "PACKEDCOLOR".

Unfortunately, there is no *comprehensive* standard definition for 3D files. The Inventor toolkit can be used to create structures that Blender will never be able to display in their entirety and Blender contains file structures that Inventor cannot display.

This problem is very likely to remain unsolved for the coming years, which means that it will remain impractical to work with a number of different 3D editors in an integrated fashion. Most organisations skirt around the problem with customised converters written by in-house programmers.

Compatibility issues will receive a high priority in design efforts for future Blender versions.

VIDEOSCAPE FILES

The Videoscape format, which is much simpler than Inventor and can even be typed in by hand, is derived from the very first 3D package for the Amiga, "VIDEOSCAPE3D". It was probably also the first commercial 3D package that could be run on a 'home' computer, as it was called in those days.

NeoGeo has worked with this standard 'object' format since it came out. Anyone with a basic knowledge of Basic or C can create good procedural designs using it. Blender is still able to read and write these files. It does require a minor change to the way in which colours are used, however. Instead of the limited 32 colormap entries, normal 24 bit colours can now be specified.

Four 'dialects' have also been developed for Videoscape. They are described below.

Due to the fact that each Videoscape file is only able to describe a single object, an extra number code is added to the file name.

For example: when the file "RT.2B1" is read, Blender also looks for the file "RT.2B12". If the file exists, it is read and Blender goes on to "RT.2B13" etc..

In Videoscape files, blank lines can be entered to make the files more readable. Use of comments is not permitted.

1. THE STANDARD MESH

```
3DGI
number
X Y Z
X Y Z
...
nr index1 index2 index3 ... colour
nr index1 index2 index3 ... colour
...

3DGI: the identification code
number: the total number of vertices
X Y Z: three numbers separated by spaces.

nr: the number of vertices in a face
index1, index2, ... the number of the vertex, the number of the first vertex is '0'
colour: 24 bit BGR hexadecimal; thus 0x0000FF for red.
```

Example of a standard 'Plane':

```
3DGI
4
1.0 1.000000 0.000000
1.000000 1.000000 0.000000
1.000000 1.000000 0.000000
-1.000000 1.000000 0.000000
4 0 3 2 1 0xcccccc
```

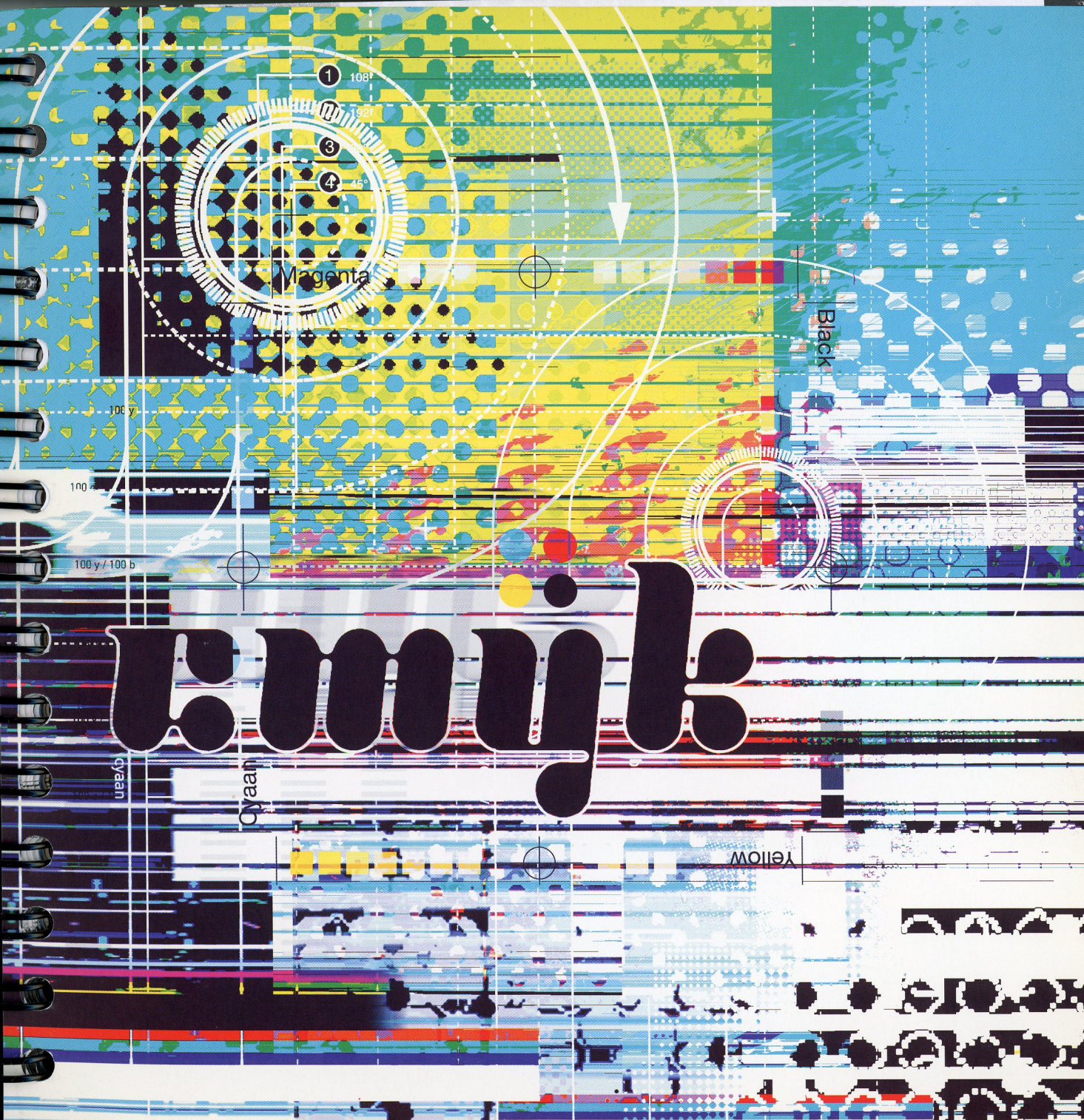
2. THE MESH WITH VERTEX COLOURS

```
GOUR
number
X Y Z colour
X Y Z colour
...
nr index1 index2 index3 ...
nr index1 index2 index3 ...
...
```

The codes are the same as those in the 3DGI version.

Example of a 'Plane' with four colours:

```
GOUR
4
1.0 1.000000 0.000000 0x0000FF
1.000000 -1.000000 0.000000 0x00FF00
1.000000 -1.000000 0.000000 0xFF00
1.000000 1.000000 0.000000 0x00FFFF
4 0 3 2 1
```



3. LAMPS

```
3DG2
number
type
spotsiz spotblend
R G B E
X Y Z
VecX VecY VecZ
```

3DG2: the code.
number: the total number of lamps in the file.

The following block must be repeated for each lamp.

type: '0' is lamp, '1' is spot lamp, '2' is Sun.

spotsiz, spotblend: the size of the spot

beam (in degrees), and the intensity of the beam.

R G B E: separated by spaces, the colour and Energy of the lamp.

X Y Z: separated by spaces, the 3D coordinate.

VecX VecY VecZ: separated by spaces, the lamp vector.

Example of two lamps, one normal and one spot lamp:

```
3DG2
2
0
0.0 0.0
1.0 1.0 1.0 1.0
-4.0 4.0 2.0
0.0 0.0 0.0
1
45.0 0.5
0.5 1.0 1.0 1.0
2.0 3.0 4.0
0.4 0.3 0.6
```

4. CURVE AND SURFACE

This file can be used to describe both Curves and Surfaces. The file is relatively complex and is described in three parts.

The header is:

```
3DG3
type
number
ext1 ext2
Matrix[4][4]
```

3DG3: the code
type: '5' is a surface, other numbers are Curves

number: the number of curves or surfaces

ext1 ext2: the extrude numbers, also indicated for Surfaces. These are *not* floats, these numbers are divided by 500.

Matrix[4][4]: four lines of four floats separated by spaces. This is the right-handed 4*4 matrix that is used to determine the position, rotation and size.

Per curve or surface, this block:

```
type
pntsu pntsv
resolu resolu
orderu orderu
flagu flagu
```


type: '0' is Poly, '1' is Bezier, '4' is Nurbs. Add '8' to this if the Curve is 2D.

pntsu pntsv: the number of vertices in the U and V direction

resolu resolu: the resolution in the U and V direction

orderu orderu: the order in the U and V direction

flagu flagu: the cyclic 'flag' in the U and V direction. '1' is cyclic.

For the Bezier curve type, this block 'pntsu' times:

X Y Z
X Y Z
X Y Z
h1 h2

X Y Z: the three *handle* vertices
h1 h2: the *handle* codes: '0' is Free, '1' is Auto, '2' is Vector, '3' is Aligned.

For other curve types or for Surfaces, this block 'pntsu x pntsv' times:

X Y Z W

X Y Z W: the coordinate and weight of the vertices.

For other curve types or for Surfaces, this block:

u1 u2 u3 ...
v1 v2 v3 ...

u1 u2 u3: the *knots* in the U direction
v1 v2 v3: only for Surfaces: the *knots* in the V direction.

The number of knots is a very precise value. If the value is not correct, the file will not be properly loaded. This is the calculation for the total:

$\text{order} + \text{pnts} + (\text{order} - 1) * [\text{flag} \& 1]$.

If necessary, a series of increasing numbers can be filled in here. Use the "MAKE KNOTS" buttons to set the correct values.

Example for a Bezier Curve:

30G3
0
1
0 0
1.0 0.0 0.0 0.0
0.0 1.0 0.0 0.0
0.0 0.0 1.0 0.0
0.0 0.0 0.0 1.0

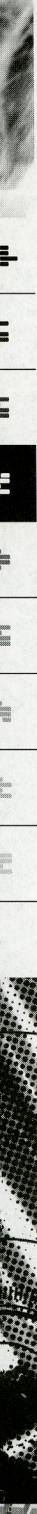
9
2 0
32 0
0 0
0 0
-3.0 1.0 0.0
-2.0 0.0 0.0
1.0 1.0 0.0
3 3
1.0 1.0 0.0
2.0 0.0 0.0
3.0 1.0 0.0
3 3

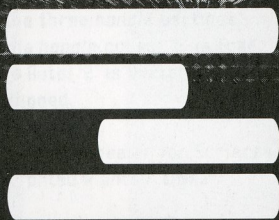
Example for a Nurbs Surface:

30G3
5
1
0 0
1.0 0.0 0.0 0.0
0.0 1.0 0.0 0.0
0.0 0.0 1.0 0.0
0.0 0.0 0.0 1.0

4
4 2
32 32
4 2
0 0
-2.0 0.0 0.0 1.0
1.0 1.0 0.0 1.0
2.0 0.0 0.0 1.0
3.0 1.0 0.0 1.0
-2.0 0.0 1.0 1.0
1.0 1.0 1.0 1.0
2.0 0.0 1.0 1.0
3.0 1.0 1.0 1.0

1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0
1.0 2.0 3.0 4.0





Part 8

Reference: the windows and hotkeys

Blender Windows - General Introduction

This section describes the general functions of the mouse and keyboard, both of which work uniformly throughout the Blender interface. Each Blender window also offers a number of specific options. These options are described in the following sections.

→078

THE MOUSE

Each time you place the mouse cursor over the edge of a Blender window, the mouse cursor changes shape. When this happens, the following mouse keys are activated:

LeftMouse (hold-move)

Drag the window edge horizontally or vertically. The window edge always moves in increments of 4 pixels, making it relatively easy to move two window edges so that they are precisely adjacent to each other, thus joining them.

MiddleMouse

A PupMenu prompts for confirmation: "OK? Split". At this point, Blender allows you to indicate the exact split point or to cancel "split" by pressing `esc`. "Split" divides the window into two windows, creating an exact copy of the original window.

RightMouse

A PupMenu prompts for confirmation: "OK? Join". Windows with a shared edge are joined, if possible.

Blender window headers offer the following extra options in combination with mouse keys:

LeftMouse on the header

The entire Blender window pops to the foreground.

CTRL+LeftMouse on the header

The entire Blender window pops to the background.

MiddleMouse (hold-move) on the header

If the Blender window is not wide enough to display the entire header, the MiddleMouse key can be used to horizontally shift the header.

RightMouse on the header

"OK? Switch header". The header can be moved to the top or the bottom of the Blender window.

WINDOW HOTKEYS

Certain window managers also use the following hotkeys.

`ALT-CTRL` can be substituted for `CTRL` to perform the functions described below.

CTRL+LEFTARROW

Go to the previous Screen.

CTRL+RIGHTARROW

Go to the next Screen.

CTRL+UPARROW or CTRL+DOWNARROW

Maximise the window or return to the previous window display size.

SHIFT+F4

Change the window to a DataView

SHIFT+F5

Change the window to a 3DWindow

SHIFT+F6

Change the window to an IpoWindow

SHIFT+F7

Change the window to a ButtonsWindow

SHIFT+F8

Change the window to a SequenceWindow

SHIFT+F9

Change the window to an OopsWindow

SHIFT+F10

Change the window to an ImageWindow

SHIFT+F11

Change the window to an ImageSelect

UNIVERSAL HOTKEYS

The following HotKeys work uniformly in all Blender windows:

ESCKEY

This key always cancels Blender functions without changes.

or: FileWindow DataView and ImageSelect: back to the previous window type.

or: the RenderWindow is *pushed* to the background.

SPACEKEY

Start the Toolbox.

→083

TABKEY

Start or quit EditMode.

F1KEY

Loads a Blender file. Changes the window to a FileWindow

SHIFT+F1

Loads as a Library from a Blender file. Changes the

window to a FileWindow making Blender files accessible as a directory (partially disabled in V 1.5).

→074

F2KEY

Writes a Blender file. Change the window to a FileWindow.

F3KEY

Writes a picture (if a picture has been rendered). The window becomes a FileWindow

F4KEY

Displays the LampButtons (if a ButtonsWindow is available).

F5KEY

Displays the MaterialButtons (if a ButtonsWindow is available).

F6KEY

Displays the TextureButtons (if a ButtonsWindow is available).

F7KEY

Displays the AnimButtons (if a ButtonsWindow is available).

F8KEY

Displays the WorldButtons (if a ButtonsWindow is available)

F9KEY

Displays the EditButtons (if a ButtonsWindow is available).

F10KEY

Displays the DisplayButtons (if a ButtonsWindow is available)

F11KEY

Push or pop the render window.

F12KEY

Start the Render

LEFTARROW

Go to the previous frame.

SHIFT+LEFTARROW

Go to the first frame.

RIGHTARROW

Go to the next frame.

SHIFT+RIGHTARROW

Go to the last frame.

UPARROW

Go forward 10 frames.

DOWNARROW

Go back 10 frames.

ALT+A

Change the current Blender window to Animation Playback mode. The cursor changes to a counter.

ALT-SHIFT+A.

The current window, *plus* all 3DWindows go into Animation Playback mode.

ALT+E

Start or leave EditMode.

IKEY

Insert Key menu. This menu differs from window to window.

JKEY

Toggle the render buffers. Blender allows you to retain two different rendered pictures in memory.

NKEY

Number buttons. These buttons differ depending on the type of Blender window. Numeric information for the active selection can be visualised and specified using these buttons.

CTRL+O

Open file. This key combination allows you to load Blender files without opening the FileWindow.

QKEY

"OK? Quit Blender". This key closes Blender. "Blender quit" is displayed in the console if Blender is properly closed.

ALT-CTRL+T

TimerMenu. This menu offers access to information about drawing speed. The results are displayed in the console.

CTRL+U

"OK? Save User defaults" The current settings (windows, objects, etc.), including UserMenu settings are written to the file \$HOME/.B.blend.

CTRL+W

Write file. This key combination allows you to write the Blender file without opening a FileWindow.

ALT+W

Write Videoscape file. Changes the window to a FileWindow.

→164

CTRL+X

Erase All. Everything is erased and released. The \$HOME/.B.blend file is reloaded.

The InfoWindow



INFOHEADER



WindowType (IconSli)

As in all Blender window headers, the first button allows you to configure the window type.

→078

The next two groups of DataButtons can be used to manage *Screens* and *Scenes*.

→070



Screen Browse (MenuBut)

Allows you to select a different Screen from a list. The option "Add New" creates an exact copy of the current Screen. The copy is 'invisible': only the name on the adjacent button changes.

HotKey for next or previous Screen:

(CTRL-)ALT+ARROWLEFT OR

(CTRL-)ALT+ARROWRIGHT

SCR: (TextBut)

Assign a new and unique name to the current Screen and then adds the new name to the list in alphabetical order

Delete Screen (But)

"Delete Current Screen?" The current Screen is deleted and released.



Scene Browse (MenuBut)

Select a different scene from a list. This button is blocked in EditMode.

"Add New" displays a PupMenu with four options:

"Empty": create a completely empty scene.

"Link Objects": all Objects are linked to the new scene. The *layer* and *selection flags* of the Objects can be configured differently for each Scene.

"Link ObData": duplicates Objects only ObData linked to the Objects, e.g. Mesh and Curve, are not duplicated.

"Full Copy" everything is duplicated.

SCE: (TextBut)

Assigns a new and unique name to the current Scene and places the new name in the list in alphabetical order

Delete Scene (But)

"OK? Delete Current Scene" This button deletes and releases the current Scene without deleting Objects.

Objects without users are not written to a file.

The information text

www.blender.nl V138 Fra:1 Ve:4 Fa:1 Ob:2-1 La:0 Mem:1.30M Time:00:00.00 (0.00) Plane

The standard text is:

www.blender.nl: the location at which the software can be obtained.

V 1.50: the version. This manual is applicable to the V1.5x series only.

Fra: the current frame number

- Ve: 4 Fa: 1: the number of *vertices* and *faces* in the current 3DWindow. If in doubt, use PAD-9 to count the *vertices* and *faces* again.
- Ob: 0-1: the number of *selected* Objects and the *total* number of Objects in the current 3DWindow.
- La: 1: the number of lamps in the current 3Dwindow.
- Mem: 1.9M: the amount of memory in use in Megabytes, not including fragmented memory.
- Time: 00.01.56 (00:44): the pure rendering time for the last picture rendered in minutes/seconds/hundredths of a second and the actual extra rendering time in seconds/hundredths of a second. Excessive swap time or poor file accessibility can cause high 'extra rendering time'.
- Plane (or similar): the name of the *active* Object.

Changes in EditMode:

- Ve: 0-4 Fa: 0-1: The first numeric value is the number of *selected* vertices, the second is the *total* number of vertices. The numeric values for the second variable apply to *faces* and have the same significance.

During and after rendering:

The values for the totals are changed to reflect the rendered picture. These values may be different from the totals displayed in the 3DWindow.

loaded from the file \$HOME/.B.blend each time Blender is started.

Personal settings cannot be written to a file other than .B.blend.

The HotKey CTRL-U can be used to overwrite the file .B.blend.

Fonts:/usr/people/trace/font/

Render:/render/

Textures:/pics/textures/

Font: (TextBut)

The directory from which Blender retrieves vector fonts for Font Objects.

Render: (TextBut)

The directory to which Blender renders by default.

Texture: (TextBut)

The directory from which Blender retrieves pictures for textures and texture plugins.

SeqPlugin (TextBut)

The directory from which Blender retrieves Sequence plugins.

Auto Temp Save

Time: 4

Dir:/usr/tmp/

Load Temp

Auto Temp Save (TogBut)

Blender can save 'temp' files at regular intervals as a temporary backup or as extra protection against disasters. The files are identical to Blender files saved in the normal manner. If Blender is in EditMode when this function is used, only the original Data are saved, without saving the Data with which you are working.

USERMENU

The UserMenu allows you to configure personal settings. These settings are automatically

Fonts:/usr/people/trace/font/		Render:/render/		Textures:/pics/textures/		SeqPlugin:/pics/blender/plugin/seq/	
Auto Temp Save	Time: 4	Versions: 1		Scene Global	NoCapsLock	Viewmove	Grab Grid
Dir:/usr/tmp/	Load Temp	TrackBall	FS Collums	Mat on Obj	Size Grid	Rot Grid	
				Dupli Mesh	Curve	Text	Lamp
				Surf	MBall	Ipo	Material
				Texture			

Blender saves 'temp' files in the specified 'temp' directory with the name "<process-id>.blend". This results in unique names for all 'temp' files, allowing multiple Blenders to simultaneously write 'temp' files on the same computer. When Blender is closed down, the file is renamed "quit.blend", making it easy to retrieve work in progress if the user inadvertently quits Blender. Blender writes files very quickly, which means that waiting time is kept to a minimum, allowing the user to continue working within a split second after saving files of 1-2 Mb.

Time: (NumBut)

The time interval in minutes required to write 'temp' files.

Dir: (TextBut)

The directory to which 'temp' files are written. Do not use network drives if possible. Instead, use a local directory on your own workstation.

Load Temp (But)

This button, the Blender disaster' button, loads the most recently saved 'temp' file.

Versions: 1

Versions (NumBut)

This option also allows you to work with Blender more safely. When this option is activated, files that are overwritten by Blender are assigned a new file name with a version number.

For example:

If 'Versions' has a value of 2 and the file 'rt.blend' is being written:
rt.blend2 is deleted
rt.blend1 is renamed to rt.blend2
rt.blend is renamed to rt.blend1
rt.blend is rewritten.

Scene Global

NoCapsLock

Viewmove

TrackBall

FS Columns

Mat on Obj

Scene Global (TogBut)

Each Screen can display a different Scene. In cases involving numerous active Screens, this can be somewhat confusing. This option can be used to display the current Scene in all Screens.

NoCapsLock (TogBut)

This button can be used to deactivate the CapsLock key. This button only applies to the TextButton.

ViewMove (TogBut)

By default, MiddleMouse causes a rotation in the 3Dwindow and SHIFT+MiddleMouse causes a translation. The option "ViewMove" toggles the key code.

TrackBall (TogBut)

There are two methods to rotate a 3DView with MiddleMouse. The TrackBall method offers more freedom than the other option, the *turn table* method.

FS Columns (TogBut)

This button activates column mode drawing in the FileWindow.

Mat on Obj (TogBut)

By default, Blender assumes that Materials are linked to the ObData when new Objects are created. This option toggles between the default and the alternative, which links Materials directly to the Object.

Grab Grid

Size Grid

Rot Grid

By default, the following *limitors* apply to *grabbing*, *rotating* and *scaling*:

- (no key): fluid change
- SHIFT: finer control
- CTRL: large grid steps
- SHIFT-CTRL: small grid steps

The following alternatives can also be used:

- (no key): large grid steps
- SHIFT: small grid steps
- CTRL: fluid change
- SHIFT-CTRL: finer control

Grab Grid (TogBut)

Toggle to the alternative *limitors* for translations.

Size Grid (TogBut)

Toggle to the alternative *limitors* for scaling.

Rot Grid (TogBut)

Toggle to the alternative *limitors* for rotation.

Dupli Mesh

Curve

Text

Lamp

Material

Surf

MBall

Ipo

Texture

Duplication / Linking presets

One of Blender's most advanced features is its use of object-oriented functions. Blender allows you to reuse (i.e. link) data blocks to construct compact and efficient structures.

→066

The most commonly use of links is when activating the *duplicate* commands:

- ALT+D: create a copy of the selected Objects, reusing (i.e. linking to) all other data, including Meshes and Materials.
- SHIFT+D: create a copy of the selected Objects, using these button settings to determine whether links to other data are created or duplicates of other data are created.

AD FILE Free: 81.105 Mb Files: (0) 72 (0.000) 8.600 Mb



Abstract

ly-
d fi

nam



	4 096	rwX	rwX	r-X	ton	10:43	15-Apr-98
	4 096	rwX	rwX	r-X	ton	18:52	02-Jul-98
man	31	rwX	rwX	r-X	ton	16:49	11-Jan-98
tex	4 096	rwX	rwX	r-X	ton	15:01	10-Apr-98
bat_keyblend	266 376	rw-	rw-	r--	ton	21:21	18-Jan-98
hevelcurve.blend	28 608	rw-	rw-	r--	ton	21:24	14-Jan-98
camera.anim.blend	36 924	rw-	rw-	r--	ton	21:53	14-Jan-98
deformtest.blend	39 520	rw-	rw-	r--	ton	22:39	14-Jan-98
flaretest.blend	26 116	rw-	rw-	r--	ton	17:48	14-Jan-98
flytest.blend	206 608	rw-	rw-	r--	ton	17:47	14-Jan-98
logo_shadow.blend	38 016	rw-	rw-	r--	ton	17:43	14-Jan-98
lostride3.blend	550 920	rw-	rw-	r--	ton	16:33	13-Apr-98
parent_rob.blend	238 132	rw-	rw-	r--	ton	22:17	14-Jan-98
README	8 837	rw-	rw-	r--	ton	10:43	15-Apr-98
mtoscopes.blend	31 292	rw-	rw-	r--	ton	14:27	15-Jan-98
sequence.blend	32 632	rw-	rw-	r--	ton	19:44	16-Jan-98
spider2.blend	83 080	rw-	rw-	r--	ton	20:51	14-Apr-98
vertexpaint.blend	81 960	rw-	rw-	r--	ton	17:44	14-Jan-98

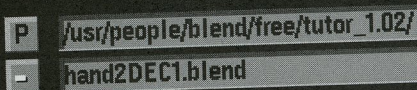
The FileWindow is generally called up to read and write files. However, you can also use it to manage the entire Unix file system. It also provides a handy overview of the internal Blender structure.

There are 4 'modes' for the FileWindow

- FileManager: the standard mode.
- FileSelect: the FileHeader shows the action to be performed (Load, Save, etc.).
- DataSelect: display the Blender data system as files.
- DataBrowse: like DataSelect, but now as an alternative to a PupMenu.

The FileWindow is optimised for reuse. The second and subsequent times the same directory is called up, the Unix file system is not re-read. This saves considerable time, but can sometimes cause confusion if other processes have written files to the directory (the directory is always read again after Blender writes to it).

If you have any doubts about the validity of the current display, press DOTKEY.



P (But)

Displays the parent directory. You can also use: PKEY.

DirName: (TextBut)

This button displays the current directory. You can also create a new directory. When you leave the button (with LeftMouse OR ENTER), you will be asked: "OK? Make dir".

Preset Directories (MenuBut)

The file \$HOME/.Bfs contains a number of presets that are displayed in this menu.

If a file is read or written, the directory involved is temporarily added to the menu.

FileName: (TextBut)

The file name can be entered here.

This button can also be used to select files using wildcards.

Example: enter *.tga', then press ENTER. All files with the extension 'tga' are selected.

FILESELECT

Blender commands such as F1 (read file) and F2 (write file) invoke a FileWindow in FileSelect mode. This mode is used to indicate a single file, i.e. the file in the FileName button. Press ENTER to initiate the action, e.g. read a Blender file. Use ESC to cancel the action.

The FileManager functions also work in FileSelect mode. These only work on *selected* files.

Standard functions in this window:

LeftMouse

Indicates the *active* file. Its name is placed in the FileName button.

MiddleMouse

Indicates the *active* file and closes the FileWindow with the message OK.

RightMouse

Select files. For functional reasons, a RightMouse Select here does not indicate the *active* file!

ENTER / PADENTER

Closes the FileWindow, returns with an OK message.

ESC

Closes the FileWindow with no further action.

PAGEDN

Scrolls down one page.

PAGEUP

Scrolls up one page.

HOMEKEY

Scrolls to the first file.

ENDKEY

Scrolls to the last file.

PADPLUSKEY / EQUALKEY

Automatic file-number increase. If there is a number in the active file name (such as `rt01.tga`), this number is incremented by one. This is quite handy when reading or writing sequential files.

PADMINUSKEY / MINUSKEY

Automatic file number decrease (see previous description).

SLASHKEY

Make the current directory the root directory: `"/`

PERIODKEY

Re-read the current directory.

EKEY

For the *active* file: start the Unix command: `$WINEEDITOR "file"` For viewing text files.

IKEY

For the *active* file: start the Unix command: `$IMAGEEDITOR "file"` For viewing or Editing images.

READ LIBRARIES

Blender allows you to read in, append, or *link* (parts of) other files. The link feature is not yet available in Version 1.5.

→074

If 'Load Library' is selected **SHIFT+F1** the FileSelect appears in a special mode. Blender files are now highlighted as directories. They are accessible as a directory as well; they then display a situation in much the same way as DataView displays the internal Blender structure.

Now you can select any number of blocks you wish using **RightMouse** and append them to the current structure with a simple **ENTER**. The complete 'underlying' structure is included: thus, if an Object is selected, the associated lpo, ObData, Materials and Textures are included as well. If a Scene is selected, the entire Scene is appended to the current structure, Objects and all.

You can specify how you want this to be appended in the FileSelect Header:

Append

Link

Append (RowBut)

External blocks become a normal part of the current structure, and thus of the file as well, if the file is saved. Appending a second time causes the entire selection to be added again in its entirety. Since block names must be unique, the name of the appended blocks will differ slightly from the existing name (only the number in the name).

Link (RowBut)

This is the 'normal' use of Libraries. The specified blocks are added to the current structure here as well, but Blender remembers the fact that they are Library blocks. When the file is saved, only the name of the Library block

and the name of the file from which the blocks were copied are saved, thus keeping the original file compact. When you read the file again, Blender reads the original file and then reads all the Library blocks from the other file(s). The names of the files and the blocks must not be changed, however. Blender keeps track of what Library blocks have already been read. Appending the same blocks twice has no consequences.

FILEMANAGER

The FileManager function only works with selected files. The active file does not play a role here.

Most of these commands do expect two FileWindows to be open. Commands such as RKEY (remove) and TKEY (touch) also work with a single Window.

Note: When we say *files* here, we also mean directories.

AKEY

Select/deselect all files.

BKEY

Backup files to the other FileWindow. This allows files to be copied without changing the file date.

CKEY

Copy files to the other FileWindow. (Unix: cp -r) read.

LKEY:

Link files to the other FileWindow. (Unix: ln)

MKEY

Move files to the other FileWindow. (Unix: mv)

RKEY

Remove files. For safety's sake: only empty directories. (Unix rm -f)

SHIFT+R

Remove files recursively. This deletes the entire contents of directories. (Unix: rm -rf)

TKEY

Update modification times of files. (Unix: touch)

DATAVIEW AND DATABROWSE

P	Object/				
-	lampcam.016				
		deur.049	1	lampquad.002	1
		deur.050	1	lampquad.003	1
		deur.051	1	lampquad.004	1
		deur.052	1	lampquad.005	1
		deur.053	1	lampquad.024	1
		deur.054	1	lampquad.025	1
		deur.055	1	lampquad.026	1
		deur.056	1	lampquad.027	1
		deur.057	1	lampquad.028	1
		deur.058	1	lampquad.029	1
		deur.059	1	pad	1
		Lamp.003	1	pad.007	1
		Lamp.004	1	pad.008	1
		Lamp.005	1	pad.010	1
		Lamp.006	1	pad.011	1
		Lamp.007	1	Plane.004	1
		Lamp.013	1	Plane.005	1
		Lamp.014	1	Plane.006	1
		Lamp.015	1	Plane.007	1
		Lamp.016	1	Plane.008	1
		Lamp.017	1	Plane.021	1
		Lamp.018	1	Plane.022	1
		lampcam	1	Plane.023	1
		lampcam.008	1	Plane.024	1
		lampcam.009	1	Plane.025	1
		lampcam.010	1	Plane.026	1
		lampcam.011	1	Plane.027	1
		lampcam.012	1	Plane.028	1
		lampcam.016	1	Plane.029	1
		lampcam.018	1	Plane.030	1
		lampquad	1	Plane.031	1
		lampquad.001	1	Plane.032	1

The DataView window can be invoked with SHIFT+F4. It allows you to view the entire internal Blender structure as a file system.

Each DataBlock is listed a file name. They are sorted by type in directories.

Currently the functions are limited to:

Select Objects by name. Click **RightMouse** on the file name. A **LeftMouse** click on a name makes the Object active. Note that an activated Object can also reside in a hidden *layer* Setting and deleting Fake Users. (Press F). A Fake User ensures that a DataBlock is always saved in a file, even if it has no users. Link Materials (**CTRL+L**). The links to *selected* Materials are all replaced by a link to the active Material (**LeftMouse**, in the **FileName** button).

This link option will be expanded to other DataBlock types.

PupMenus that contain more than 24 items and are thus unmanageably large, are replaced by the DataBrowse windows.

Standard functions in this window:

LeftMouse

Display the active DataBlock. This is placed in the **FileName** button.

MiddleMouse

Display the active DataBlock and close the DataBrowse with an OK message.

ENTER / PADENTER

Close the DataBrowse, return with an OK message.

ESC

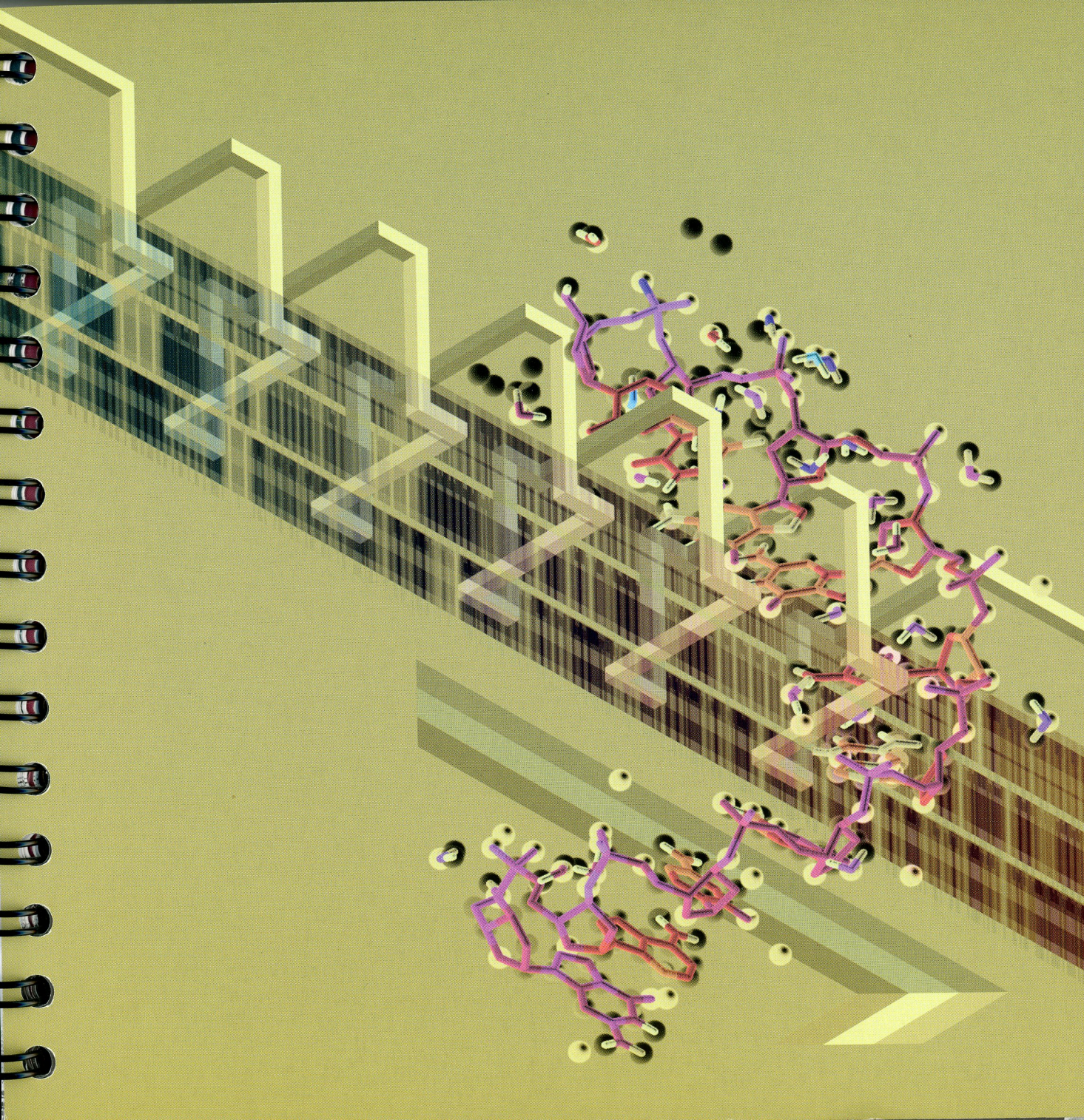
Close the DataBrowse with no further action.

PAGEDN

Scroll down one page.

PAGEUP

Scroll up one page.





The 3DWindow



3DHEADER



WindowType (IconSli)

As with every window header, the first button allows you to set the window type.



Full Window (IconTog)

Maximise the window, or return it to its original size; return to the old screen setting.

Hotkey: (ALT)-CTRL-UPARROW

→078

Home (IconBut)

All Objects in the visible layers are displayed completely, centered in the window. Hotkey: HOMEKEY.



Layers (TogBut)

These 20 buttons show the available layers. In fact, a layer is nothing more than a *visibility flag*. This is an extremely efficient method for testing Object visibility. This allows the user to divide the work functionally.

For example: Cameras in layer 1, temporary Objects in layer 20, lamps in layers 1, 2, 3, 4 and 5, etc.

All hotkey commands and tools in Blender take the layers into account. Objects in 'hidden' layers are treated as *unselected*.

Use LeftMouse for the buttons, SHIFT-LeftMouse for *extend select* layers.

Hotkeys: 1KEY, 2KEY, etc. 0KEY, MINUSKEY, EQUALKEY for layers 1, 2, 3, 4, etc.

Use ALT+(1KEY, 2KEY, ... 0KEY) for layers 11, 12, ... 20.

Here, as well, use SHIFT-Hotkey for *extend select*.

Lock (TogBut)

Every 3DWindow has its own layer setting and active Camera. This is also true for a Scene: here it determines what layers - and what camera - are used to render a picture.

The *lock* option links the layers and Camera of the 3DWindow to the Scene and vice versa: the layers and Camera of the Scene are linked to the 3DWindow.

This method passes a layer change directly to the Scene and to all other 3DWindows with the "Lock" option ON.

Turn the "Lock" OFF to set a layer or Camera *exclusively* for the current 3DWindow.

All settings are immediately restored by turning the button back ON.



LocalView (IconTog)

LocalView allows the user to continue working with complex Scenes. The currently selected Objects are taken separately, centered and displayed completely. The use of 3DWindow layers is temporarily disabled. Reactivating this option restores the display of the 3DWindow in its original form.

If a picture is rendered from a LocalView, only the Objects present are rendered *plus* the visible lamps, according to the layers that have been set. Activating a new Camera in LocalView does not change the Camera used by the Scene.

Normally, LocalView is activated with the hotkey
PAD_SLASH.



View Mode (IconSli)

A 3DWindow offers 3 methods for 3D display:
Orthonormal Blender offers this method from
every view not just from the X, Y or Z axes.
Perspective. You can toggle between
orthonormal and *perspective* with the
hotkey PAD_5.
Camera. This is the view as rendered.
Hotkey: PAD_0.

View Direction (IconSli)

These pre-sets can be used with either
ortho or *perspective*. Respectively, these
are the:

Top View hotkey PAD_7

Front View hotkey PAD_1

Right View, hotkey PAD_3

The hotkeys combined with SHIFT give
the opposite view direction. (Down
View Back View Left View)

Draw Mode (IconSli)

Set the drawing method.

Respectively:

BoundingBox. The quickest method,
for animation previews, for
example.

WireFrame.

Solid. Zbuffered with the
standard OpenGL lighting.

Hotkey: ZKEY, this toggles
between WireFrame and Solid.
Shaded. This is as good an
approach as is possible to the
manner in which Blender
renders with Gouraud
shading. It displays the
situation from a single

frame of the Camera. Hotkey: SHIFT+Z. Use CTRL+Z to
force a recalculation.
Textured. This method is not supported in this
version.

Objects have their own Draw Type, independent
of the window setting (see
EditButtons→DrawType). The rule is that the
minimum DrawMode is displayed.



View Move (IconBut, click-hold)

Move the mouse for a *view* translation. This is
an alternative for SHIFT+MiddleMouse.

View Zoom (IconBut, click-hold)

Move the mouse vertically to zoom in and out of
the 3DWindow. This is an alternative for
CTRL+MiddleMouse.



These buttons determine the manner in which
the Objects (or vertices) are *rotated* or *scaled*.

Around Center (IconRow)

The midpoint of the *boundingbox* is the center of
rotation or scaling Hotkey: COMMAKEY.

Around Median (IconRow)

The median of all Objects or vertices is the
center of rotation or scaling

Around Cursor (IconRow)

The 3DCursor is the midpoint of rotation or
scaling. Hotkey: DOTKEY.

Around Individual Centers (IconRow)

All Objects rotate or scale around their own
midpoints.

In EditMode: all vertices rotate or scale around
the Object midpoint.

EditMode

(IconTog)

EditMode (IconTog)

This button starts or terminates EditMode.

Hotkey: TAB or ALT+E.

→026

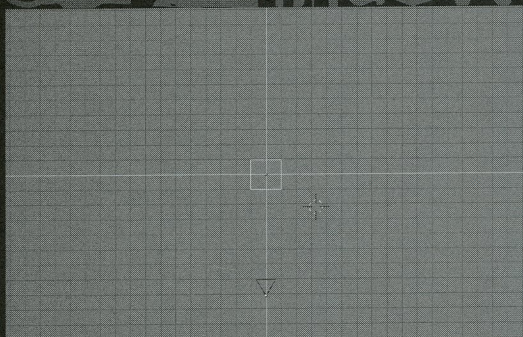
VertexPaint (IconTog)

This button starts or terminates

VertexPaintMode. Hotkey: VKKEY.

→155

3DWINDOW



The standard 3DWindow has:

- A grid. The dimensions (distance between the gridlines) and resolution (number of lines) can be set with the ViewButtons. This grid is drawn as infinite in the presets of *ortho* ViewMode (Top, Front, Right view). In the other *views*, there is an finite 'floor'. Many Blender commands are adjusted to the *dimension* of the grid, to function as a standard 'unit'. Blender works best if the total 'world' in which the user operates continually falls more or less within the total grid floor (whether it is a space war or a logo animation).
- Axes in colour codes. The reddish line is the X axis, the green line is the Y axis, the blue line is the Z axis.

In the Blender universe, the 'floor' is normally formed by the X and Y axes. The height and 'depth' run along the Z axis.

- A 3D cursor. This is drawn as a black cross with a red/white striped circle. A *LeftMouse* click moves the 3DCursor. Use the *SnapMenu* (SHIFT-S) to give the 3DCursor a specific location. New Objects are placed at the 3D cursor location.
- Layers (visible in the header buttons). Objects in 'hidden' layers are not displayed. All hotkey commands and tools in Blender take the layers into account: Objects in the 'hidden' layers are treated as *not* selected. See the following paragraph as well.
- ViewButtons. Separate variables can be set for each 3Dwindow, e.g for the *grid* or the *lens*. Use the SHIFT+F7 hotkey or the WindowType button in the 3DHeader. The ViewButtons are explained in detail elsewhere in this manual.

→231

THE MOUSE

The mouse provides the most direct access to the 3DWindow. Below is a complete overview:

LeftMouse

Position the 3DCursor.

CTRL+LeftMouse

In EditMode: create a new vertex.

LeftMouse (click-hold-draw)

These are the Gestures. Blender's gesture recognition works in three ways:

- Draw a straight line: start *translation* mode (Grabber)
- Draw a curved line: start *rotation* mode.
- Draw a V-shaped line: start *scaling* mode.

MiddleMouse (click-hold)

Rotate the direction of view of the 3DWindow. This can be done in two ways (can be set in the UserMenu):

The *trackball* method.

In this case, where in the window you start the mouse movement is important. The rotation can be compared to rotating a ball as if the mouse grasps and moves a tiny miniscule point on a ball and moves it.

If the movement starts in the middle of the window the *view* rotates along the horizontal and vertical window axes. If the movement begins at the edge of the window the *view* rotates along the axis perpendicular to the window.

The *turntable* method.

A horizontal mouse movement always results in a rotation around the global Z axis. Vertical mouse movements are corrected for the view direction, and result in a combination of (global) X and Y axis rotations.

SHIFT+MiddleMouse (click-hold)

Translate the 3DWindow. Mouse movements are always corrected for the view direction.

CTRL+MiddleMouse (click-hold)

Zoom in/out on the 3DWindow

RightMouse

Select Objects or (in EditMode) vertices. The last one selected is also the *active* one. This method guarantees that a maximum of 1 Object and 1 vertex are always selected. This selection is based on graphics (the wireframe).

SHIFT+RightMouse

Extend select Objects or (in EditMode) vertices. The last one selected is also the *active* one. Multiple Objects or *vertices* may also be selected. This selection is based on graphics too (the wireframe)

CTRL+RightMouse

Select Objects on the Object-centers. Here the wireframe drawing is not taken into account. Use this method to select a number of identical Objects in succession, or to make them active.

SHIFT-CTRL+RightMouse

Extend select Objects. The last Object selected is also the *active* one. Multiple Objects can be selected.

RightMouse (click-hold-move)

Select and start *translation* mode, the Grabber. This works with all the selection methods mentioned.

→088

PAD

The numeric keypad on the keyboard is reserved for view related hotkeys. Below is a description of all the keys with a brief explanation.

PAD_SLASH

LocalView The Objects selected when this command is invoked are taken separately and displayed completely, centered in the window. See the description of 3DHeader→LocalView

PAD_STAR

Copy the rotation of the *active* Object to the current 3DWindow. Works as if this Object is the camera, without including the translation.

PAD_MINUS, PAD_PLUS

Zoom in, zoom out. This also works for Camera ViewMode.

PAD_DOT

Center and zoom in on the selected Objects. The view is changed in a way that can be compared to the LocalView option.

PAD_5

Toggle between *perspective* and *orthonormal* mode.

PAD_9

Force a complete recalculation (of the animation systems) and draw again.

PAD_0

View from the current *camera*, or from the Object that is functioning as the *camera*.

CTRL+PAD_0

Make the *active* Object the *camera*. Any Object can be used as the camera. Generally, a Camera Object is used. It can also be handy to let the Lamp function temporarily as a camera when directing and adjusting it.

ALT+PAD_0

Revert to the previous *camera*. Only Camera Objects are candidates for 'previous camera'.

PAD_7

Top View. (along the negative Z axis, Y up)

SHIFT+PAD_1

Down View. (along the positive Z axis, Y up)

PAD_1

Front View. (along the positive Y axis, Z up)

SHIFT+PAD_1

Back View. (along the negative Y axis, Z up)

PAD_3

Right View. (along the negative X axis, Z up)

SHIFT+PAD_3

Left View. (along the positive X axis, Z up)

PAD_2, PAD_8

Rotate using the *turntable* method. Depending on the view, this is a rotation around the X and Y axis.

PAD_4 PAD_6

Rotate using the *turntable* method. This is a rotation around the Z axis.

SHIFT+(PAD_2, PAD_8)

Translate up or down; corrected for the view.

SHIFT+(PAD_4, PAD_6)

Translate up or down; correct for the view.

HOTKEYS

This list contains all the hotkeys that can be used in a 3D window. EditMode hotkeys are described in the following paragraph.

HOMEKEY:

All Objects in the visible layer are displayed completely, centered in the window.

PAGEUP

Select the next Object Key. If more than one Object Key is selected, the selection is shifted up cyclically.

Only works if the AnimButtons→DrawKey is ON for the Object.

SHIFT+PAGEUP

Extend *select* the next Object Key.

PAGEDOWN

Select the previous Object Key. If more than one Object Key is selected, the selection is shifted up cyclically. Only works if the AnimButtons→DrawKey is ON for the Object.

SHIFT+PAGEDOWN

Extend select the previous Object Key.

ACCENTKEY (To the left of the 1KEY)

Select all layers.

SHIFT+ACCENTKEY

Revert to the previous layer setting

AKEY

Select All / deselect All. If one Object or vertex is selected, everything is always deselected first.

SHIFT+A

This is the AddMenu. In fact, it is the ToolBox that starts with the 'ADD' option. If Objects are added, Blender starts EditMode immediately, if possible. As long as EditMode is active, this part of the AddMenu cannot return to main mode.

CTRL+A

"Apply size/rot" The rotation and dimensions of the Object are assigned to the ObData (Mesh, Curve, etc.). At first glance, it appears as if nothing has changed, but this can have considerable consequences for animations or texture mapping.

This is best illustrated by also having the axis of a Mesh Object be drawn (EditButtons→Axis). Rotate the Object and activate

Apply. The rotation and dimensions of the Object are 'erased'

CTRL-SHIFT+A

If the active Object is automatically duplicated (see AnimButtons→DupliFrames or AnimButtons→Dupliverts), a menu asks "Make dupli's real?" This option *actually* creates the Objects.

If the active Mesh Object is deformed by a Lattice, a menu asks "Apply Lattice deform?" Now the deformation of the Lattice is assigned to the vertices of the Mesh. The Lattice deformation remains, now having a double effect.

BKEY

Border Select. Draw a rectangle with the LeftMouse; all Objects within this area are *selected*, but *not* made active. Draw a rectangle with the RightMouse to *deselect* Objects.

In orthonormal ViewMode, the dimensions of the rectangle are displayed, expressed as global coordinates, as an extra feature in the lower left corner. In Camera ViewMode, the dimensions that are to be rendered according to the DisplayButtons are displayed in *pixel* units.

SHIFT+B

Render Border This only works in Camera ViewMode. Draw a rectangle to render a smaller cut-out of the standard window frame. If the option DisplayButtons→Border is ON, a box is drawn with red and black lines.

→288

CKEY

Centre View. The position of the 3DCursor becomes the new centre of the 3DWindow.

SHIFT+C

CentreZero View. The 3DCursor is set to zero (0,0,0) and the view is changed so that all

Objects, including the 3Dcursor, can be displayed. This is an alternative for HOMEKEY.

ALT+C

Convert Menu. Depending on the active Object, a PupMenu is displayed. This enables you to convert certain types of ObData. It only converts in one direction, everything ultimately degrades to a Mesh!
The options are:

Font → Curve
MetaBall → Mesh
Curve → Mesh
Surface → Mesh

CTRL+C

Copy Menu. This menu copies information from the active Object to (other) selected Objects.

- Fixed components are:

"Copy Loc": the X,Y,Z location of the Object. If a Child is involved, this location is the relative position in relation to the Parent.

"Copy Rot": the X,Y,Z rotation of the Object.

"Copy Size": the X,Y,Z dimension of the Object.

"DrawType" : see EditButtons→DrawType.

"TimeOffs" : see AnimButtons→TimeOffs.

"Dupli": all Duplicator data from the AnimButtons

- If applicable:

"Copy TexSpace": The texture space.

"Copy Particle Settings": the complete particle system from the AnimButtons.

- For Curve Objects:

"Copy Bevel Settings": all beveling data from the EditButtons.

- Font Objects:

"Copy Font Settings": font type, dimensions, spacing.

"Copy Bevel Settings": all beveling data from the EditButtons.

- Camera Objects:

"Copy Lens": the lens value.

SHIFT+D

"Add Duplicate". The selected Objects are copied. The settings in the UserMenu (Duplication/Linking presets) determine what DataBlocks are also copied or are linked. Blender then automatically starts Grab Mode (see GKEY).

→070

ALT+D

"Add Linked Duplicate". The selected Objects are copied. The DataBlocks linked to Objects remain linked.

Blender then automatically starts Grab Mode (see GKEY).

CTRL+D

Draw the (texture) Image as wire. This option has a limited function. It can only be used for 2D compositing.

→159

ALT+E

Start / stop EditMode. Alternative hotkey: TAB.

→026

SHIFT+F

Fly Mode. Only from Camera ViewMode. The mouse cursor jumps to the middle of the window. It works as follows:

- Mouse cursor movement indicates the view direction.

- LeftMouse click (repeated): Fly faster.

- MiddleMouse click (repeated): Fly slower.

- LeftMouse+MiddleMouse: Set speed to zero.
CTRL: translation downwards (negative Z).

- ALT: translation upwards (positive Z).

- Esc: Camera back to its starting position, terminate Fly Mode.

SPACEKEY: Leave the Camera in current position, terminate Fly Mode.
(Be careful when looking straight up or down. This causes confusing turbulence.)

CTRL+F

Sort Faces. The *faces* of the *active* Mesh Object are sorted, based on the current view in the 3DWindow. The leftmost *face* first, the rightmost last.

The sequence of *faces* is important for the Build Effect (AnimButtons).

GKEY

Grab Mode. Or: the *translation* mode. This works on selected Objects and *vertices*. Blender calculates the quantity and direction of the translation, so that they correspond *exactly* with the mouse movements, regardless of the ViewMode or view direction of the 3DWindow

Alternatives for starting this mode:

RightMouse (click-hold-move)

LeftMouse (click-hold-draw) to draw a straight line.

The following options are available in *translation* mode:

Limitors:

CTRL: in increments of 1 grid unit.

SHIFT+CTRL: in increments of 0.1 grid unit.

MiddleMouse toggle:

A short *click* restricts the current translation to the X,Y or Z axis. Blender calculates which axis to use, depending on the already initiated mouse movement.

Click **MiddleMouse** again to return to unlimited translation.

ARROWKEYS:

These keys can be used to move the mouse cursor exactly 1 pixel.

Grabber can be terminated with:

LeftMouse, SPACEBAR OR **ENTER:** move to a new position.

RightMouse or **Esc:** everything goes back to the old position.

Switching mode:

GKEY: starts Grab mode again.

SKEY: switches to Size mode.

RKEY: switches to Rotate mode.

ALT+G

Clear location. The X,Y Z locations of selected Objects are set to zero.

IKEY:

Insert Object Key. A *key position* is inserted in the current frame of all *selected* Objects. A PupMenu asks what key position(s) must be added to the *lpoCurves*.

"Loc": The XYZ location of the Object.

"Rot": The XYZ rotation of the Object.

"Size": The XYZ dimensions of the Object

"LocRot": The XYZ location and XYZ rotation of the Object.

"LocRotSize": The XYZ location, XYZ rotation and XYZ dimensions of the Object.

"Layer": The layer of the Object.

"Avail" A position is only added to all the current *lpoCurves*.

"Effector": (only for *lka* Objects) the end-effector position is added.

"Mesh" "Lattice" "Curve" or "Surface": depending on the type of Object, a *VertexKey* can be added

CTRL+J

Join Objects. All *selected* Objects of the same type are added to the *active* Object. What actually happens here is that the *ObData* blocks are combined and all the selected Objects (except for the *active* one) are deleted.

This is a rather complex operation, which can lead to confusing results, particularly when working with a lot of linked data, animation curves and hierarchies.

KEY

Show (as) Keys. The DrawKey is turned ON for all selected Objects. If all of them were already ON, they are all turned OFF.

→119

SHIFT+K

A PupMenu asks: "OK? Show and select all keys". The DrawKey is turned ON for all selected Objects, and all Object-keys are selected. This function is used to enable *transformation* of the entire animation system.

LKEY

Local Menu.
(disabled)

SHIFT+L

Select Links menu. This menu enables you to select Objects that are linked to the same DataBlock as the *active* Object. Use this function to obtain an overview of the sometimes quite complicated linked structures Blender can create.

"Object lpo": all Objects with the same Object lpo are selected.

"Object Data": all Objects with the same ObData (Mesh, Curve, etc.) are selected.

"Current Material": all Objects with the same *active* Material are selected (this is the Material in MaterialButtons).

"Current texture": all Objects with the same *active* Texture are selected (this is the Texture in MaterialButtons).

CTRL+L

Make Links menu. This menu is used to copy links, such as those between the *active* Object and selected Objects. Only the links (the

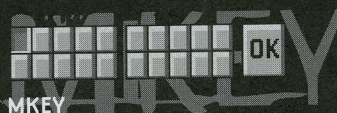
references) are copied; the contents of the DataBlocks involved are not changed. The result of this operation is easy to see in the OopsWindow.

"To Scene": the selected Objects are also linked to another Scene. A second PupMenu asks the user to specify a Scene. Objects that appear in more than one Scene are displayed with a blue null point.

"Object lpo": all selected Objects are given a link to the Object lpo of the *active* Object.

"Mesh data", "Curve data", "Font data", etc.: all selected Objects are given a link to the ObData of the *active* Object. Objects must be the same type!

"Materials": all selected Objects are given links that are identical to the Material(s) of the *active* Object. The entire Material situation is copied. If the *active* Object does not have any Materials, all Material links for the selected Objects are erased.



MKEY

Move to Layer. This hotkey calls up a menu that can be used to view or change the layer settings of all the *selected* Objects. If the selected Objects have different layers, this is 'OR'ed in the menu display.

Use ESC to exit the menu. Press the "OK" button or ENTER to change the layer setting.

The hotkeys (ALT+)(1KEY, 2KEY, ... - 0KEY) work here as well (see 3DHeader).

Active Object

LocX: -7.39	OK
LocY: -3.01	
LocZ: 0.00	
RotX: 0.96	
RotY: -1.01	
RotZ: -0.88	
SizeX: 1.47	
SizeY: 1.47	
SizeZ: 1.47	

NKEY

These buttons allow the user to enter numbers for the position, rotation and dimensions of the *active* Object.

Use **esc** to exit the menu without changing the Object. Press the "OK" button or **ENTER** to assign the changes to the Object.

ALT+O

Clear Origin. The 'Origin' is erased for all Child Objects, which causes the Child Objects to move to the exact location of the Parent Objects.

CTRL+P

Make Parent. The *active* Object becomes the Parent of the selected Objects. All transformations of the Parent are now passed on to the Children. This allows you to create complex hierarchies. As part of the 'Make Parent' operation, an *inverse* of the Parent transformation is calculated and stored in

the Child Object. This *inverse* may make it seem as if all transformations remained unchanged after Make Parent was executed. Depending on the type of Object, special Parent relationships can be selected.

→089

ALT+P

Clear Parent. All selected Child Objects are unlinked from the Parents. A PupMenu asks you to make a selection:

"Clear Parent": the selected Child Objects are unlinked from the Parent. Since the transformation of the Parent disappears, this can appear as if the former Children themselves are transformed.

" and keep transform": the Child Objects are unlinked from the Parent, and an attempt is made to assign the current transformation, which was determined in part by the Parent, to the (former Child) Objects.

"Clear Parent inverse" The inverse matrix of the Parent of the selected Objects is erased. The other Child Objects remain linked to the Parent. This gives the user complete control over the hierarchy

→089

RKEY

Rotate mode. Works on selected Objects and *vertices*. In Blender a rotation is *always* a rotation perpendicular to the screen, regardless of the view direction or ViewMode.

The degree of rotation is *exactly* linked to the mouse movement. Try moving around the rotation midpoint with the mouse.

The rotation midpoint is determined by the state of the 3DHeader→Around buttons.

Alternatives for starting *rotation* mode:

LeftMouse (click-hold-draw): draw a C-shaped curve.

The following options are available in *rotation* mode:

- Limitors:
 - CTRL: in increments of 5 degrees.
 - SHIFT: finer control.
 - SHIFT+CTRL: finer control with increments of 1 degree.
- MiddleMouse toggle:
A short *click* switches the current rotation axis, which is perpendicular to the screen, with the two other axes, which are vertical and horizontal on the screen. The mouse movements here follow the two rotation axes. Click MiddleMouse again to return to a rotation axis perpendicular to the screen.
- ARROWKEYS:
These keys move the mouse cursor exactly 1 pixel.
Switching mode:
RKEY: starts Rotate mode again.
SKEY: switches to Size mode.
GKEY: switches to Grab mode.
- Rotation mode can be terminated with:
 - LeftMouse, SPACEBAR OR ENTER: move to a new location.
 - RightMouse OR ESC: everything returns to the former state.

ALT+R

Clear Rotation. The X,Y,Z rotations of selected Objects are set to zero.

SKEY

Size mode or *scaling* mode. Works on selected Objects and *vertices*.

The degree of *scaling* is *exactly* linked to the mouse movement. Try to move from the (rotation) midpoint with the mouse.

The midpoint is determined by the settings of the 3DHeader→Around buttons.

- Alternatives for starting scaling mode:
 - LeftMouse (click-hold-draw) draw a V-shaped line.

The following options are available in *scaling* mode:

- Limitors:
 - CTRL: in steps of 0.1.
 - SHIFT+CTRL: in steps of 0.01.
- MiddleMouse toggle:
A short *click* restricts the scaling to the X, Y or Z axis. Blender calculates the appropriate axis based on the already initiated mouse movement.
Click MiddleMouse again to return to free scaling.
- ARROWKEYS:
These keys move the mouse cursor exactly 1 pixel.
- XKEY
Make the horizontal scaling negative; this is also called an X-flip.
- YKEY
Make the vertical scaling negative; this is also called a Y-flip.
- Switching mode:
 - SKEY: starts Size mode again.
 - RKEY: switches to Rotate mode.
 - GKEY: switches to Grab mode.
- Terminate Size mode with:
 - LeftMouse, SPACEBAR OR ENTER: move to a new location.
 - RightMouse OR ESC: everything returns to its previous state.

ALT+S

Clear Size. The X,Y,Z dimensions of selected Objects are set to 1.0.

SHIFT+S

Snap Menu. This menu offers a number of options for precisely specifying the position of Objects or vertices.

"Sel → Grid": all selected Objects or vertices are moved to the closest grid position.

"Sel → Curs": all selected Objects or vertices are moved to the 3DCursor position.

"Curs → Grid": the 3DCursor is moved to the closest grid position.

"Curs → Sel": the 3DCursor is moved to the selected Object. If there are multiple Objects or vertices, the 3DCursor is set to the midpoint. The 3DHeader→Around buttons determine what type of midpoint is meant here.

Tip: use the PupMenu hotkeys IKEY, ZKEY, etc. to select the options.

TKEY

Texture space mode. The position and dimensions of the texture space for the selected Objects can be changed in the same manner as described above for Grab and Size mode. To make this visible, the *drawing flag* EditButtons→TexSpace is set ON.

A PupMenu asks you to select: "Grabber" or "Size"

Faster hotkeys: T-1 or T-2.

CTRL+T

Make Track. All selected Objects are given a rotation *constraint* directed at the *active* Object. The type of rotation and which axis is up and which one is directed at the Track Object are specified in the AnimButtons.

ALT+T

Clear Track. The *tracking* is turned off for all selected Objects. A PupMenu allows you to maintain the current *tracking* as the rotation in the Objects.

UKEY

Single User Menu. Use this menu to manage the (multi) user structure. These operations only work on selected Objects and the DataBlocks that are linked to them.

"Object": if other Scenes also have a link to this Object, the link is deleted and the Object is copied. The Object now only exists in the current Scene. The links *from* the Object remain unchanged.

"Object & ObData": Similar to the previous command, but now the ObData blocks with multiple links are copied as well. All selected Objects are now present in the current Scene only and each has a unique ObData (Mesh, Curve, etc.).

"Object & ObData & Materials+Tex": Similar to the previous command, but now Materials and Textures with multiple links are also copied. All selected Objects are now unique. They have unique ObData and each has a unique Material and Texture block.

"Materials+Tex": Only the Materials and Textures with multiple links are copied.

VKEY

Start or exit VertexPaint Mode.

→155

ALT+V

Object-Image Aspect. This hotkey sets the X and Y dimensions of the selected Objects in relation to the dimensions of the Image Texture they have. Use this hotkey when making 2D Image compositions and multi-plane designs to quickly place the Objects in the appropriate relationship with one another

ALT+W

Write Videoscape. The selected Mesh Object is saved as an ASCII file. Use the FileWindow to enter a file name. Blender adds the extension ".obj" to the Videoscape file.

→164

XKEY

Erase Selected. All selected Objects are deleted from the Scene and, if they are not linked to

other Scenes, they are released. The ObData and other Linked DataBlocks remain.

ZKEY

DrawMode Solid ON/OFF. This is Zbuffered with the standard OpenGL lighting.

SHIFT+Z

DrawMode Shaded ON. This drawing mode, which is Zbuffered and Gouraud shaded, approaches the way in which Blender renders as closely as possible. It shows the situation from a single Camera frame.

CTRL+Z

Turn DrawMode Shaded ON and force a recalculation of DrawMode Shaded.

EDITMODE HOTKEYS - GENERAL

TABKEY / ALT+E

This button starts and stops EditMode.

→026

AKEY

Add Menu. In EditMode, this menu can only be used to add the primitives of the type of Object that is in EditMode.

BKEY

Border Select. In EditMode, this function only works on the vertices. It works as described in the previous section.

BKEY-BKEY

Circle Select. If you press BKEY a second time after starting Border Select, Circle Select is invoked. This mode selects vertices with LeftMouse and deselects vertices with MiddleMouse. Use PAD_PLUS or PAD_MINUS to adjust the circle size. Leave Circle Select with RightMouse or ESC.

DKEY

Add Duplicate. The selected vertices (faces, curves) are copied. Grab mode starts immediately thereafter.

NKEY

NumberMenu. In EditMode Mesh, Curve, Surface: The location of the active vertex is displayed.

PKEY

seParate. All selected vertices, edges, faces and curves are removed from the EditMode and placed in a new Object. This operation is the opposite of Join (CTRL+J).

CTRL+P

"Make Vertex Parent". Select 1 or 3 vertices from a Mesh, Curve or Surface. Now this Object becomes the Vertex Parent of the selected Objects.

If only 1 vertex is selected, only the *location* of this vertex determines the Parent transformation; the rotation and dimensions of the Parent do not play a role here.

If three vertices are selected, it is a 'normal' Parent relationship in which the 3 vertices determine the rotation and location of the Child *together*. This method produces interesting effects with Vertex Keys.

In EditMode, other Objects can be selected with CTRL+RightMouse.

CTRL+S

Shear. This operation enables you to make selected forms 'slant'. This always works via the horizontal screen axis.

UKEY

(For Font Objects: ALT+U)

Reload Original Data. When starting EditMode, the original ObData block is saved. This option enables you to restore the previous situation.

By continually leaving EditMode while working (TAB-TAB) you can refresh this undo buffer'

WKEY

Specials PupMenu. A number of *tools* are included in this PupMenu as an alternative to the EditButtons. This makes the buttons accessible as *shortcuts*, e.g EditButtons→Subdivide is also 'WKEY, IKEY'

SHIFT+W

Warp. Selected vertices can be bent into curves with this option. It can be used to convert a plane into a tube or even a sphere.

The centre of the circle is the 3DCursor
The mid-line of the circle is determined by the horizontal dimensions of the selected vertices. When you start, everything is already bent 90 degrees. Moving the mouse up or down increases or decreases the extent to which *warping* is done. By zooming in/out of the 3Dwindow you can specify the maximum degree of *warping*.

The CTRL limiter increments warping in steps of 5 degrees.

EDITMODE MESH HOTKEYS

EKEY

Extrude Selected. "Extrude" in EditMode transforms all the selected *edges* to *faces*. If possible, the selected faces are also duplicated.
Grab mode is started directly after this command is executed.

→092

FKEY

Make Edge/Face. If 2 vertices are selected, an *edge* is created. If 3 or 4 vertices are selected, a *face* is created.

SHIFT+F

Fill selected. All selected vertices that are bound by *edges* and form a closed polygon are filled with triangular *faces*. Holes are automatically taken into account. This operation is 2D; various layers of polygons must be filled in succession.

ALT+F

Beauty Fill. The edges of all the selected triangular faces are switched in such a way that equally sized faces are formed. This operation is 2D; various layers of polygons must be filled in succession.
The Beauty Fill can be performed immediately after a Fill.

HKEY

Hide Selected. All selected vertices and faces are temporarily hidden.

SHIFT+H

Hide Not Selected: All *non-selected* vertices and faces are temporarily hidden.

ALT+H

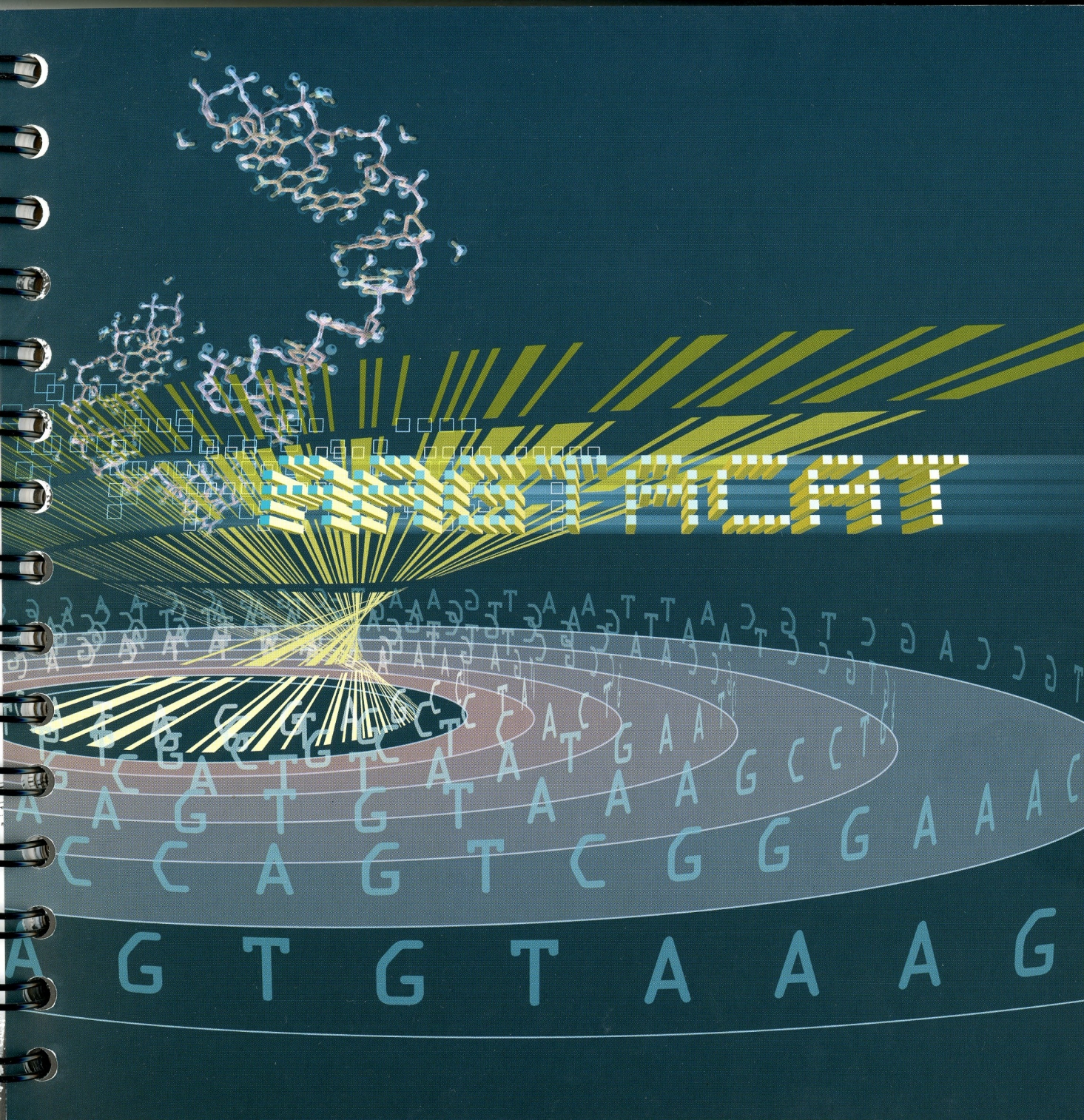
Reveal. All temporarily hidden vertices and faces are drawn again.

LKEY

Select Linked. If you start with an *unselected* vertex near the mouse cursor this vertex is selected, together with all vertices that are linked with edges to it.

SHIFT+L

Deselect Linked. If you start with a *selected* vertex, this vertex is deselected, together with all vertices that are linked with edges to it.





CTRL+L

Select Linked Selected. Starting with *all* selected vertices, all vertices connected to them are selected too.

CTRL+N

Calculate Normals Outside. All normals from selected faces are recalculated and consistently set in the same direction. An attempt is made to direct all normals 'outward'.

SHIFT-CTRL+N

Calculate Normals Inside. All normals from selected faces are recalculated and consistently set in the same direction. An attempt is made to direct all normals 'inward'.

CTRL+T

Make Triangles. All selected faces are converted to triangles.

XKEY

Erase Selected. A PupMenu offers the following options:

"Vertices": all vertices are deleted. This includes the edges and faces they form.

"Edges": all edges with both vertices selected are deleted. If this 'releases' certain vertices, they are deleted as well. Faces that can no longer exist as a result of this action are also deleted.

"Faces": all faces with all their vertices selected are deleted. If any vertices are 'released' as a result of this action, they are deleted.

"All": everything is deleted.

"Edges and Faces": all selected edges and faces are deleted, but the vertices remain.

"Only Faces": all selected faces are deleted, but the edges and vertices remain.

YKEY

Split. This command 'splits' the selected part of a Mesh without deleting faces. The split

parts are no longer bound by *edges*. Use this command to control *smoothing*.

Since the split parts have vertices at the same position, selection with *LKEY* is recommended.

EDITMODE CURVE HOTKEYS

CKEY

Set the selected curves to cyclic or turn cyclic off. An individual curve is selected if at least one of the vertices is selected.

EKEY

Extrude Curve. A vertex is added to the selected end of the curves. Grab mode is started immediately after this command is executed.

FKEY

Add segment. A segment is added between two selected vertices at the end of two curves. These two curves are combined into 1 curve.

HKEY

Toggle Handle *align/free*. Toggles the selected Bezier *handles* between *free* or *aligned*.

SHIFT+H

Set Handle *auto*. The selected Bezier *handles* are converted to *auto* type.

CTRL+H

Calculate Handles. The selected Bezier curves are calculated and all *handles* are assigned a type.

→098

LKEY

Select Linked. If you start with an *non-selected* vertex near the mouse cursor, this vertex is selected together with all the vertices of the same curve.

SHIFT+L

Deselect Linked. If you start with a *selected* vertex, it is deselected together with all the vertices of the same curve.

TKEY

Tilt mode. Specify an extra axis rotation, i.e. the *tilt*, for each vertex in a 3D curve.

ALT+TKEY

Clear Tilt. Set all axis rotations of the selected vertices to zero.

VKEY

Vector Handle. The selected Bezier *handles* are converted to *vector* type.

CTRL+W

Switch direction. The direction of the selected curves is reversed. This is for Curves that are used as *paths*!

XKEY

Erase Selected. A PupMenu offers the following options:

"Selected": all selected vertices are deleted.

"Segment": a curve segment is deleted. This only works for single segments.

Curves can be split in two using this option. Or use this option to specify the cyclic position within a cyclic curve.

"All": delete everything

EDITMODE FONT HOTKEYS

A complete list of all key combinations for special characters can be found elsewhere in this manual.

→104

SHIFT+TAB

The ASCII code for TAB.

RIGHTARROWKEY

Move text cursor 1 position forward

SHIFT+RIGHTARROWKEY

Move text cursor to the end of the line.

LEFTARROWKEY

Move text cursor 1 position backwards.

SHIFT+LEFTARROWKEY

Move text cursor to the start of the line

DOWNARROWKEY

Move text cursor 1 line forward

SHIFT+DOWNARROWKEY

Move text cursor to the end of the text.

UPARROWKEY

Move text cursor 1 line back.

SHIFT+UPARROWKEY

Move text cursor to the beginning of the text

ALT+UP/DOWNARROWKEY

Next or previous ASCII value.

ALT+U

"Reload Original Data" (undo)

When EditMode is started, the original text is saved. You can restore this original text with this option.

ALT+V

Paste text. The text file "/tmp/cutbuffer" is inserted at the cursor location.

EDITMODE SURFACE HOTKEYS

CKEY

Toggle Cyclic menu. A PupMenu asks if selected surfaces in the 'U' or the 'V' direction must be cyclic. If they were already cyclic, this mode is turned off

EKEY

Extrude Selected. Extrude makes *surfaces* of all the selected *curves*, if possible. Only the edges of surfaces or loose curves are candidates for this operation.

Grab mode is started immediately after this command is completed.

FKEY

Add segment. A segment is added between two selected vertices at the ends of two curves. These two curves are combined into 1 curve.

LKEY

Select Linked. If you start with an *non-selected* vertex near the mouse cursor, this vertex is selected together with all the vertices of the same curve or surface.

SHIFT+L

Deselect Linked. If you start with a *selected* vertex, this vertex is deselected together with all vertices of the same curve or surface.

SHIFT+R

Select Row. Starting with the last selected vertex, a complete row of vertices is selected in the 'U' or 'V' direction. Selecting "Select Row" a second time with the same vertex switches the 'U' or 'V' selection.

CTRL+W

sWitch direction. The direction of the selected curves is reversed.

XKEY

Erase Selected. A PupMenu offers the following choices:

"Selected": all selected vertices are deleted.

"All": delete everything.

IKA HOTKEYS

TABKEY

Turns the calculation of 'Inverse Kinematics' on or off.

EKEY

Extrude Ika. An extra Limb is added to the selected Ikas. Grab mode is started immediately thereafter.

CTRL+K

Make skeleton. A Skeleton is created (anew) in the *active* Ika Object. All selected Ikas and Empty Objects become part of the Skeleton. To indicate that an Ika has a Skeleton, the 'end-effector' is now drawn in blue (instead of yellow).

CTRL+P

Make Parent. If the Parent is an Ika, a PupMenu offers three options:

"Use vertex": one of the vertices from the Ika *chain* becomes the Parent. Only the location transformation is passed on to the Child Objects. Indicate which vertex with the NumberButton that pops up.

"Use limb": one of the limbs of the Ika becomes the Parent. Indicate which Limb with the NumberButton that pops up.

"Use skeleton": all Ikas and Emptys that make up the skeleton, become the Parent and can distort the Child Objects. Distortion only works on Objects that are *directly* parented to the Skeleton.

VERTEXPAINT HOTKEYS

SHIFT+K

All vertex colours are erased; they are changed to the current drawing colour

UKEY

Undo. This undo is 'real' Pressing Undo twice returns you to the previous situation.

WKEY

"Shared Vertexcol": The colours of all faces that share vertices are blended.

The ButtonsWindow



BUTTONSHEADER



WindowType (IconSli)

As with every window header, the first button enables you to set the window type.



Full Window (IconTog)

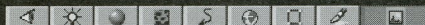
Maximise the window, or return to the previous window display size; return to the old screen setting.

HotKey: (ALT)-CTRL-UPARROW

Home (IconBut)

The optimal *view* settings for the current window are restored.

HotKey: HOMEKEY.



The following series of buttons determine the type of ButtonsWindow.

In order, they are:

ViewButtons. (IconRow)

The 3DWindow settings for the window.

LampButtons (IconRow)

The settings of the active Lamp Object.

HotKey: F4.

MaterialButtons (IconRow)

The settings of the active Material linked by the active Object. HotKey F5.

TextureButtons (IconRow)

The settings of the active Texture. This can be the Texture for a Material, Lamp or World.

HotKey: F6.

AnimButtons (IconRow)

The (animation) settings of the active Object, including Particle Effects. HotKey: F7.

WorldButtons (IconRow)

The settings of the World linked by the current Scene. HotKey: F8.

EditButtons (IconRow)

The settings and EditMode options of the ObData linked by the *active* Object. HotKey: F9.

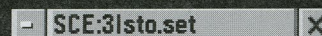
PaintButtons (IconRow)

The VertexPaint options. Only for an *active* Mesh Object.

DisplayButtons (IconRow)

The settings of the Scene. HotKey: F10.

Later sections deal with each button in detail.

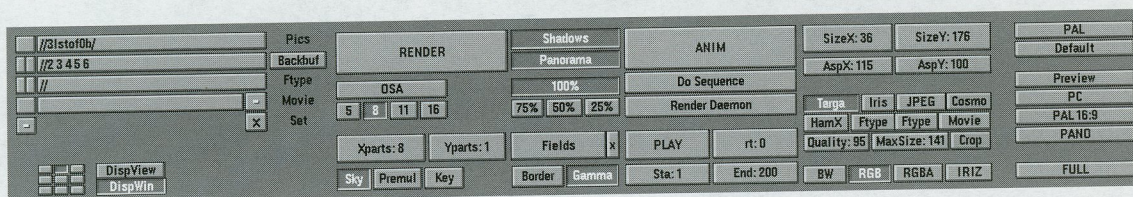


This group of DataButtons is drawn based on the type of ButtonsWindow and the availability of the data to be visualised. The header for the DisplayButtons is shown here as an example.

→070

Scene Browse (MenuBut)

Choose a different scene from the list of available scenes.



SCE: (TextBut)

Give the current Scene a new and unique name.

Delete Scene (But)

"OK? Delete Current Scene"

11

Current Frame. (NumBut)

The current frame number is displayed as a fixed part of the ButtonsHeader

BUTTONSWINDOW

A buttonsWindow offers a number of extra facilities:

MiddleMouse (hold-move)

If space is available, this can be used to horizontally or vertically move the view.

CTRL+MiddleMouse (hold move)

Within certain limits, a ButtonsWindow can be enlarged or reduced.

PAD_PLUS

Zoom in.

PAD_MINUS

Zoom out.

HOMEKEY

The optimal view settings for the current window are restored.

If there is only one 3Dwindow in the current Screen, the PAD commands for the 3DWindow also work in the ButtonsWindow

The IpoWindow



IPOHEADER



WindowType (IconSli)

As with every window header, the first button enables you to set the window type.

Full Window (IconTog)

Maximise the window or return it to the previous window display size; return to the old screen setting.

HotKey: (ALT-)CTRL+UPARROW

Home (IconBut)

All visible curves are displayed completely, centered in the window. HotKey: HOMEKEY.



IpoKeys (IconTog)

This is a drawing mode for the animation curves in the IpoWindow (the IpoCurves). Yellow vertical lines are drawn through all the vertices of the curves. Vertices of different curves at the same location in 'time' are joined together and can easily be selected, moved, copied or deleted.

This method adds the ease of traditional key framing to the animation curve system.

For Object-Ipos, these IpoKeys can also be drawn and transformed in the 3DWindow. Changes in the 3D position are processed in the IpoCurves immediately.

→119



Ipo Type

Depending on the active Object, the various Ipo systems can be specified with these buttons.

Object Ipo (IconRow)

Settings, such as the location and rotation, are animated for the active Object. All Objects in Blender can have this Ipo block.

Material Ipo (IconRow)

Settings of the active Material are animated for the active Object.

A NumBut is added as an extra feature. This button indicates the number of the active Texture *channel*. Eight Textures, each with its own mapping, can be assigned per Material. Thus, per Material-Ipo, 8 curves in the row "OfsX, OfsY, ...Var" are available.

Speed Ipo (IconRow)

If the active Object is a *path* Curve, this button can be used to set the speed.

Lamp Ipo (IconRow)

If the active Object is a Lamp, this button can be used to animate light settings.

World Ipo (IconRow)

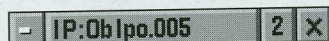
Used to animate a number of settings for the WorldButtons.

VertexKey Ipo (IconRow)

If the active Object has a VertexKey, the keys are drawn as horizontal lines. Only one IpoCurve is available to interpolate between the Keys.

Sequence Ipo (IconRow)

The active Sequence Effect can have an IpoCurve.



The DataButtons can be used to control the Ipo blocks themselves.

Ipo Browse (MenuBut)

Choose another Ipo from the list of available Ipos. The option "Add New" makes a complete copy of the current Ipo. This is not visible; only the name in the adjacent button will change.

Only Ipos of the same type are displayed in the menu list.

IP: (TextBut)

Give the current Ipo a new and unique name. After the new name is entered, it appears in the list, sorted alphabetically

Users (NumBut)

If this button is displayed, there is more than one user for the Ipo block. Use the button to make the Ipo "Single User"

Unlink Ipo (IconBut)

The current Ipo is unlinked.



Copy to Buffer (IconBut)

All selected IpoCurves are copied to a temporary buffer

Paste from Buffer (IconBut)

All selected *channels* in the IpoWindow are assigned an IpoCurve from the temporary buffer. The rule is: the sequence in which they are copied to the buffer is the sequence in which they are pasted.

A check is made to see if the number of IpoCurves is the same.



Extend mode Constant (IconBut)

The end of selected IpoCurves are horizontally extrapolated.

Extend mode Direction (IconBut)

The ends of selected IpoCurves continue extending in the direction in which they end.

Extend mode Cyclic (IconBut)

The full length of the IpoCurve is repeated cyclically



View Zoom (IconBut, click-hold)

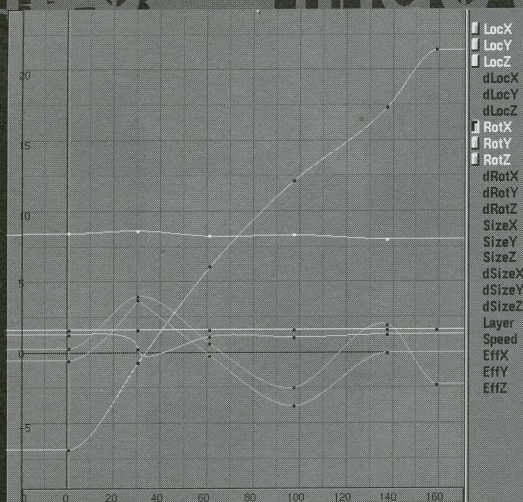
Move the mouse horizontally or vertically to zoom in or out on the IpoWindow. This is an alternative for CTRL+MiddleMouse.

View Border (IconBut)

Draw a rectangle to indicate what part of the IpoWindow should be displayed in the full window

IpoWindow

LocX



The IpoWindow shows the contents of the Ipo block. Which one depends on the Ipo Type specified in the header.

The standard IpoWindow has a grid with the time expressed horizontally in frames and vertical values that depend on the *channel*.

There are 2 sliders at the edge of the IpoWindow. How far the IpoWindow is zoomed in can be seen on the sliders, which can also be used to move the view.

The right-hand part of the window shows the available *channels*.

To make it easier to work with rotation-IpoCurves, they are displayed in degrees (instead of in radians). The vertical scale relation is: 1.0 'Blender unit' = 10 degrees.

In addition to the IpoCurves, the VertexKeys are also drawn here. These are horizontal lines; the yellow line visualises the *reference* Key.

Each *channel* can be operated with two buttons:

IpoCurve Select (TogBut)

This button is only displayed if the *channel* has an IpoCurve. The button is the same colour as the IpoCurve. Use the button to select IpoCurves. Multiple buttons can be (de)selected using SHIFT+LeftMouse.

Channel Select (TogBut)

A *channel* can be selected whether there is an IpoCurve or not. Only IpoCurves of selected channels are drawn. Multiple *channels* can be (de)selected using SHIFT+LeftMouse.

THE MOUSE

LeftMouse (hold-draw)

These are the Gestures. Blender's gesture recognition works in two ways here:

- Draw a straight line: start *translation* mode (Grabber)
- Draw a V-shaped line: start *scaling* mode.

CTRL+LeftMouse

Create a new vertex. These are the rules:

- There is no Ipo block (in this window) and one *channel* is selected: a new IpoBlock is created along with the first IpoCurve with one vertex.
- There is already an Ipo block, and a *channel* is selected without an IpoCurve: a new IpoCurve with one vertex is added
- Otherwise a new vertex is simply added to the selected IpoCurve.
- This is *not* possible if multiple IpoCurves are selected or if you are in EditMode.

MiddleMouse (hold-move)

Depending on the position within the window:
On the *channels*; if the window is not high enough to display them completely the visible part can be shifted vertically.

On the sliders; these can be moved. This only works if you are zoomed in.

The rest of the window; the *view* is translated.

CTRL+MiddleMouse (hold-move)

Zoom in/out on the IpoWindow. You can zoom horizontally or vertically using horizontal and vertical mouse movements.

RightMouse

Selection works the same here as in the 3DWindow: normally one item is selected. Use **SHIFT** to expand or reduce the selection (*extend select*).

If the IpoWindow is in IpoKey mode, the IpoKeys can be selected.

If at least 1 of the IpoCurves is in EditMode, only its vertices can be selected.

VertexKeys can be selected if they are drawn (horizontal lines)

The IpoCurves can be selected.

RightMouse (click-hold-move)

Select and start *translation* mode, i.e. the Grabber. The selection can be made using any of the four selection methods discussed above.

SHIFT+RightMouse

Extend the selection.

THE HOTKEYS

PAD_MINUS, PAD_PLUS

Zoom in, zoom out.

PAGEUP

Select the next IpoKey. If more than one IpoKey is selected, the selection is shifted cyclically.

SHIFT+PAGEUP

Extend select the next IpoKey.

PAGEDOWN

Select the previous IpoKey. If more than one Object Key is selected, the selection is shifted cyclically.

SHIFT+PAGEDOWN

Extend select the previous IpoKey.

HOMEKEY

All visible curves are displayed completely centered in the window.

TABKEY

All selected IpoCurves go into or out of EditMode. This mode allows you to transform individual vertices.

AKEY

Select All / deselect All. If 1 item is selected, first everything is deselected. Placing the mouse cursor above the *channels*, (de)selects all channels where there is a curve.

BKEY

Border select. Draw a rectangle with the **LeftMouse**; all items that fall within this rectangle are *selected*. Draw a rectangle with the **RightMouse** to *deselect*.

CKEY

If one vertex or one IpoKey is selected, the current *frame* number is placed in this position.

SHIFT+DKEY

Duplicate Ipo. All selected vertices or IpoKeys are copied. Then *translation* mode is started automatically.

GKEY

Translation mode (the Grabber). This works on selected curves, keys or vertices.

Alternatives for starting this mode:

- **RightMouse** (click-hold-move)
- **LeftMouse** (click-hold-draw) draw a straight line.

The following options are available in *translation* mode:

- **Limitors:**
 - **CTRL:** in increments of 1 frame or vertical unit.
 - **SHIFT+CTRL:** in increments of 0.1 frame or vertical unit.
- **MiddleMouse toggle:**
A short *click* restricts the current translation to the X or Y axis. Blender calculates which axis to use, based on the already initiated mouse movement.
Click **MiddleMouse** again to restore unlimited translation.
- **ARROWKEYS:**
With these keys the mouse cursor can be moved exactly 1 pixel.
- Grabber can be terminated with:
 - **LeftMouse, SPACEBAR** or **ENTER:** move to a new position.
 - **RightMouse** or **ESC:** everything returns to the old position.

HKEY

Toggle Handle *align* / *free*. The selected Bezier *handles* toggle between *free* and *aligned* types.

SHIFT+H

Set Handle *auto*. The selected Bezier *handles* are converted to *auto* type.

→098

IKEY

Insert Key. Vertices can be added to the visible curves in the IpoWindow.

A PupMenu asks you to make a choice:

- "**Current Frame**"; all visible curves get a vertex on the current frame.
- "**Selected Keys**"; (only in IpoKey mode) all selected IpoKeys get vertices for each visible curve, including IpoCurves that are not linked to the IpoKey.

JKEY

Join vertices. Selected vertices or IpoKeys can be joined. A PupMenu asks you to make a choice:

- "**All Selected**"; all selected vertices are replaced by a new one.
- "**Selected doubles**"; all selected vertices that are closer to each other than 0.9 *frame* are joined.

KKEY

IpoKey mode ON/OFF. If the Ipo block is Object Ipo type, the Objects are redrawn with the option DrawKey ON (see the explanation under IpoHeader).

RKEY

Recording mode. The X and Y movements of the mouse are linked to the height of the IpoCurve. Thus, this works with a maximum of two selected *channels* or IpoCurves.

The old curve is completely deleted; the new one becomes a 'linear' type. You cannot change parts of curves with *recording*.

The scale at which this happens is determined by the *view* of the IpoWindow.

A PupMenu asks you to make a choice:

"Still"; the current frame is used as the starting point.

"Play anim"; the animation starts, allowing you to see the correlation with other animation systems.

During *recording* mode, the **CTRL** key must be held down to actually start recording

Press **SPACEKEY** OR **ENTER** OR **LeftMouse** to stop *recording*. Use **ESCKEY** to undo changes.

SKEY

Scaling mode. This works on selected **IpoCurves** and vertices.

The degree of *scaling* is *precisely* linked to the mouse movement. Try to move from the (rotation) midpoint with the mouse.

In **IpoKey** mode, you can only *scale* horizontally

Alternatives for starting *scaling* mode:

LeftMouse (click-hold-draw) draw a sharp angle; a V-shaped line.

The following options are available in *scaling* mode:

Limitors:

CTRL: in increments of 0.1.

SHIFT+CTRL: in increments of 0.01.

MiddleMouse toggle:

A short *click* limits *scaling* to the X or Y axis. Blender calculates which axis to use based on the already initiated mouse movement.

Click **MiddleMouse** again to return to free *scaling*.

ARROWKEYS:

These keys allow you to move the mouse cursor exactly 1 pixel.

XKEY

Make the horizontal *scaling* negative, the X-flip.

YKEY

Make the vertical *scaling* negative, the Y-flip.

Terminate size mode with:

LeftMouse OR **SPACEBAR** OR **ENTER**: move to a new location.

RightMouse OR **ESC**: everything returns to its previous state.

SHIFT+S

Snap Menu.

"Horizontal": The selected Bezier handles are set to horizontal.

"To next": The selected handle or vertex is set to the same (Y) value as the next one.

"To frame": The selected handles or vertices are set to the exact frame values.

"To current frame": The selected handle or vertex is moved to the current frame.

TKEY

If an **IpoCurve** is selected: "**Ipo Type**" The type of selected **IpoCurves** can be changed. A **PupMenu** asks you to make a choice:

"Constant": after each vertex of the curve, this value remains constant, and is not interpolated.

"Linear": linear interpolation occurs between the vertices.

"Bezier": the vertices get a *handle* (i.e. two extra vertices) with which you can indicate the curvature of the interpolation curve.

If a **Key** is selected: "**Key Type**" The type of selected **Keys** can be changed.

"Linear": linear interpolation occurs between the **Keys**. The **Key** line is displayed as a broken line.

"Cardinal": fluent interpolation occurs between the **Keys**; this is the default.

"BSpline": extra fluent interpolation occurs between the **Keys**, four **Keys** are now involved in the interpolation calculation. Now the positions *themselves* cannot be displayed precisely however The **Key** line is shown as a broken line.

VKEY

Vector Handle. The selected Bezier *handles* are converted to *vector* type.

XKEY

Erase selected. The selected vertices, IpoKeys or IpoCurves are deleted.

If there are selected VertexKeys, they are also deleted.

The SequenceWindow



SEQUENCEHEADER



WindowType (IconSli)

As with every window header the first button allows you to set the type of window

Full Window (IconTog)

Maximise the window or return to the previous window display size; return to the previous screen setting.

Hotkey: (ALT)-CTRL+UPARROW

Home (IconBut)

All visible Sequences are completely displayed, centered in the window.

Hotkey: HOMEKEY.



DisplayImage (IconTog)

The window shows the result of the Sequences, i.e. a picture.



View Zoom

(IconBut, click-hold)

Move the mouse to zoom into or out of the SequenceWindow.

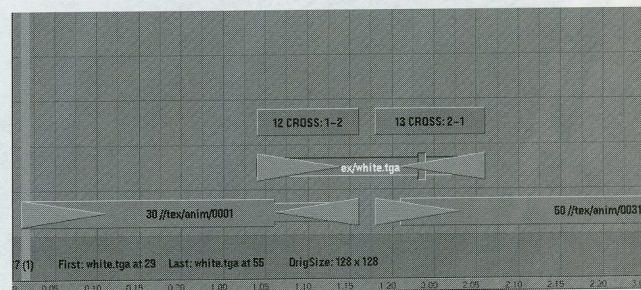
This is an alternative for

CTRL+MiddleMouse.

View Border (IconBut)

Draw a rectangle to indicate what part of the SequenceWindow should be displayed in the full window.

SEQUENCEWINDOW



The Sequences (strips) in this window are part of a Scene. They are needed for special effects when creating pictures and for montages. Each Scene has its own collection of Sequences.

In this example, the two long strips are sequences of pictures. The first strip has been lengthened; the last picture remains for half a second.

The middle image strip (white.tga) is a single picture that has been lengthened to 27 frames.

In the topmost layer there are two Cross effects. The numbers in the strips (1-2, 2-1) show what layers are crossed.

→150

The standard SequenceWindow has a grid. The time is expressed horizontally the layers vertically. In video terms, we would call these the 'machines'. The lowermost piece, layer 0, gives information about the selected *strip* as text.





The vertical green line is the current frame number

If the option `DisplayImage` is ON, this window shows the result of the Sequences. Generally two `SequenceWindows` are opened, one of which shows the image.

The result of Sequences is always shown in the resolution as specified in the `DisplayButtons` with "`SizeX`" and "`SizeY`". If the option `DisplayButtons→OSA` is ON, pictures are filtered enlarged or filtered reduced.

THE MOUSE

LeftMouse

The position of the mouse cursor becomes the current frame. If `DisplayImage` is ON, the Sequences at this position are read or calculated at this position.

MiddleMouse (hold-move)

Depending on the position within the window:

- On the sliders; this can be moved.
- The rest of the window; the *view* is translated.

CTRL+MiddleMouse (hold-move)

Zoom in/out on the `IpoWindow`. For ease of use, you can only zoom horizontally. Use the `HeaderButton` if you must zoom vertically as well.

RightMouse

Selection works the same way here as in the `3DWindow`: normally a maximum of one Sequence strip is selected. Use `SHIFT` to extend or reduce the selection (*extend select*).

RightMouse (click-hold-move)

Selects something and immediately starts *translation mode*, i.e. the Grabber.

SHIFT+RightMouse

Extend selection.

THE HOTKEYS

PAD_MINUS, PAD_PLUS

Zoom in, zoom out.

SHIFT+PAD_PLUS

Insert gap. One second is inserted at the current frame position. This only applies to strips that are totally to the right of the current frame.

ALT+PAD_PLUS

Insert gap. As above, but now 10 seconds are inserted.

SHIFT+PAD_MINUS

Remove gaps. All strips that are completely to the right of the current frame and do *not* start past the last frame are repositioned so that there is no longer an empty space.

PAD_DOT

The last selected strip is displayed completely.

HOMEKEY

All visible Sequences are displayed completely, centered in the window.

AKEY

Select All / deselect All. If 1 strip is selected, everything is first deselected.

SHIFT+A

Add sequence. A `PupMenu` asks you to make a choice. The first three are possible sources:

- "Images"; Specify with `FileSelect` (with `RightMouse select!`), what pictures will form a strip. If only 1 picture is selected, the strip is lengthened to 50 frames. Directories can also

be specified; each directory then becomes a separate strip in the SequenceWindow "Movie"; Specify with FileSelect (with LeftMouse or MiddleMouse!) what SGI movie will comprise a strip (Only SGI systems).

"Scene"; A PupMenu asks you to specify the Scene that is to be inserted as a strip. The Scene is then rendered according to its own settings and processed in the Sequence system.

→150

The following menu options are *effects* that work on pictures; two strips must be selected for this. The order of selection determines how the effects are applied.

"Cross"; a fluent transition from strip 1 to strip 2.

"GammaCross"; This is a gamma corrected cross. It provides a more 'natural' transition in which lighter parts are inserted before darker parts.

"Add"; two strips are added together
"Sub"; the second strip is subtracted from the first.

"Mul"; the strips are multiplied.

- "AlphaOver"; the second strip, with its *alpha*, is placed over the first. pictures with *alpha* are normally 32 bits.

"AlphaUnder"; the first strip is placed behind the second, with the *alpha* of the second strip.

"AlphaOverDrop"; like

"AlphaOver" but now with a drop shadow

BKEY

Border select. Draw a rectangle with the LeftMouse; all strips that fall within this rectangle are *selected*. Draw a rectangle with the RightMouse to *deselect*.

CKEY

If one of the ends of a strip is selected (the triangles), the current frame is moved to this end.

In all other cases, the Change menu is invoked. This menu allows you to change specific characteristics of the *active* strip.

If this is an Image strip:

"Change Images" The FileSelect appears and new pictures can be specified.

If this is an Effect strip:

"Switch a-b"; change the sequence of the effect.
"Recalculate"; force a recalculation of the effect.

"Cross, Gammacross, Add, "; change the type of effect.

If this is a Scene strip:

"Update Start and End"; the start and end frame of the Scene is processed in the strip again.

ALT+D

Add Duplicate. All selected strips are copied. Immediately thereafter *translation* mode is started. The images in an Image strip are reused; they take up no extra memory.

FKEY

"Set Filter" An extra Y filter can only be activated in Movie strips. This filter is for a stable video display with no flickering

GKEY

Translation mode (the Grabber). This works on selected strips or the (triangular) ends of selected strips.

Alternative for starting this mode: RightMouse (click-hold-move).

The following options are available in *translation* mode:

Limitors:

SHIFT: with finer translation.

MiddleMouse toggle:

A short *click* limits the current translation to the X or Y axis. Blender calculates which axis to

use based on the already initiated mouse movement.

Click **MiddleMouse** again to return to unlimited translation.

- **ARROWKEYS:**

These keys can be used to move the mouse cursor exactly 1 pixel.

- Grabber can be terminated with:

- **LeftMouse**, **SPACEBAR** OR **ENTER**: move to a new position.
- **RightMouse** OR **ESC**: everything returns to the old position.

MKEY

Make Meta. The selected strips are combined into a Meta strip. This only occurs if no unselected strips are linked to the selection by effects.

Use **TABKEY** to view the contents of a Meta or to leave the Meta.

Metas can be inside other Metas, and behave exactly like a normal Sequence strip. When Metas are duplicated, their contents are *not* linked!

ALT+M

Un-Meta. The selected Meta is 'unpacked' again.

SKEY

Snap menu. The PupMenu offers you a choice:

- "Sequence to frame"; the selected strips are placed with their starting point on the current frame.

XKEY

Delete Sequence. The selected strips are deleted.

The OopsWindow



OOPSHEADER



WindowType (IconSli)

As with every window header the first button allows you to set the window type.

Full Window (IconTog)

Maximise the window, or return to the previous window display size; return to the previous screen setting.

HotKey: (ALT-)CTRL+UPARROW

Home (IconBut)

All visible blocks are displayed completely centered in the window.

HotKey: HOMEKEY.



View Zoom

(IconBut, click-hold)

Move the mouse to zoom in or out of the OopsWindow. This is an alternative to CTRL+MiddleMouse.

View Border (IconBut)

Draw a frame to indicate what part of the OopsWindow should be displayed in the full window.



Visible Select (IconTog)

This row of buttons determines what types of DataBlocks must be displayed. Relations between blocks are only shown if both blocks are visible.

These are:

Lay: the *layer* setting of the Scene determines what Objects are drawn.

Scene: like ON, all Scenes present are displayed.

Object: all Objects of all visible Scenes are displayed, possibly limited by the "Lay" option.

Mesh

Curve: this is also for Surface and Font blocks.

MetaBall

Lattice

Lamp

Material

Texture

Ipo

Library

OOPSWINDOW

The Materials ("Brown" and "SeaGreen") are linked to the ObData and the Texture is linked to a Material.

→066

THE MOUSE

LeftMouse (hold-draw)

These are the Gestures. Blender's gesture recognition works in two ways here:

- Draw a straight line:
start *translation* mode.
- Draw a V-shaped line:
start *scaling* mode.

MiddleMouse (hold-move)

The *view* is translated.

CTRL+MiddleMouse (hold-move)

Zoom in or out of the OOpsWindow.

RightMouse

Select works here in the normal fashion: normally a maximum of one DataBlock is selected. Use **SHIFT** to enlarge or reduce the selection (*extend select*).

RightMouse (click-hold-move)

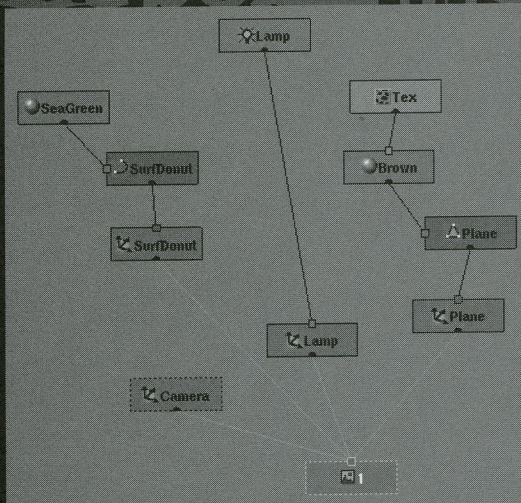
Select and start *translation* mode, the Grabber.

SHIFT+RightMouse

Extend selection.

CTRL+RightMouse

This selects and *activates* a DataBlock. This only works for Scenes and Objects.



The OOpsWindow gives a schematic overview of the current structure. Blender is based on an Object-Oriented Programming System (OOPSI). The building blocks are universal DataBlocks that are assigned relationships using *links*. The different DataBlocks in the OOpsWindow can be recognised by an icon and the colour. The DataBlocks have a clearly visible 'entrance' and 'exit' between which *link* lines are drawn.

The current Scene and the active Object have a frame with a broken line.

The functionality of the OOpsWindow is limited to visualisation. Links are created using the available HotKeys (CTRL-L in the 3DWindow) and with the DataButtons in the Headers. Selected Objects are also selected in the OOpsWindow, and vice versa.

In the accompanying example, we see the Scene at the bottom, with four linked Object blocks. The Object blocks have links to the specific ObData; such as Mesh, Surface, Lamp.

THE HOTKEYS

PAD_MINUS, PAD_PLUS

Zoom in, zoom out.

HOMEKEY

All visible blocks are displayed completely centred in the window

ONEKEY, TWOKEY, EQUALKEY

The visible *layers* of the current Scene can be set. Use **SHIFT** for *extend* select.

AKEY

Select All / deselect All. If one block is selected, everything is first deselected.

BKEY

Border select. Draw a rectangle with the **LeftMouse**; all blocks that fall within this rectangle are *selected*. Draw a rectangle with the **RightMouse** to *deselect* the blocks.

GKEY

Translation mode (the Grabber). This works on selected blocks.

Alternatives for starting this mode:

RightMouse (click-hold-move)

LeftMouse (click-hold-draw) draw a straight line.

The following options are available in *translation* mode:

- **MiddleMouse** toggle:

A short *click* restricts the current translation to the X or Y axis. Blender calculates which axis to use based on the already initiated mouse movement.

Click **MiddleMouse** again to restore unlimited translation.

ARROWKEYS:

The mouse cursor can be moved exactly 1 pixel with these keys.

Grabber terminates with:

LeftMouse OR **SPACEBAR** OR **ENTER**: move to a new position.

RightMouse OR **ESC**: everything returns to the old position.

LKEY

Select Linked Forward. All *DataBlocks* that are linked by a selected *DataBlock* are also selected. In this way the entire underlying structure can be selected, starting with a selected Scene block.

SHIFT+L

Select Linked Backward. All *users* of selected *DataBlocks* are selected. This allows you to visualise what Objects the Material uses, starting with a selected Material block.

SKEY

Scaling mode. This works on selected blocks. Only the length of the links, i.e. the distance between the *DataBlocks*, can be transformed. The degree of *scaling* corresponds *exactly* to the mouse movement. Try to move the (rotation) midpoint with the mouse.

Alternatives for starting *scaling* mode:

LeftMouse (click-hold-draw) draw a sharp angle; a V-shaped line.

The following options are available in *scaling* mode:

MiddleMouse toggle:

A short *click* restricts the *scaling* to the X or Y axis. Blender calculates which axis to use based on the already initiated mouse movement.

Click **MiddleMouse** again to return to free *scaling*.

ARROWKEYS:

These keys move the mouse cursor exactly 1 pixel.

LeftMouse OR SPACEBAR OR ENTER: move to a new location.

RightMouse OR ESC: everything returns to its previous state.

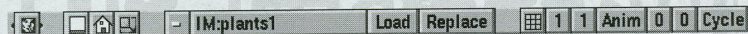
SHIFT+S

Shuffle Oops. An attempt is made to minimise the length of the *link* lines for selected DataBlocks using parsed toggling.

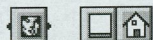
ALT+S

Shrink Oops. The length of the *link* lines for the selected DataBlocks is shortened without causing the blocks to overlap.

The ImageWindow



IMAGEHEADER



WindowType (IconSli)

As with every window header the first button allows you to set the type of window.

Full Window (IconTog)

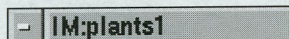
Maximise the window, or return to the previous window display size; return to the previous screen setting

HotKey: (ALT)-CTRL+UPARROW

Home (IconBut)

The picture is displayed completely centred in the window.

HotKey: HOMEKEY.



The Image blocks can be specified using the DataButtons.

Image Browse (MenuBut)

Select an Image from the list provided.

IM: (TextBut)

Give the current Image a new and unique name. After the new name is entered, it is displayed in the list alphabetically



Load (But)

Load a new Image. A FileSelect asks you to specify what file.

Replace (But)

Replace the picture in the current Image block with a new one.

IMAGEWINDOW



Images in Blender are also DataBlocks. The functionality of the ImageWindow is limited to visualisation in this version.

The use of the mouse and HotKeys is:

MiddleMouse (hold-move)

Translate the view.

PAD_MINUS, PAD_PLUS

Zoom in, zoom out.

HOMEKEY

The picture is displayed completely, centred in the window.

CTRL+N

Replace Image Names. A menu asks you to enter an old and a new file name. All file names for Images with the old name or a name which starts with corresponding characters are replaced by the new name.

This utility is especially useful for changing directories.

Example:

"old" = /data/,

"new" = /pics/textures/. The file name

"/data/marble.tga" is replaced by

"/pics/textures/marble.tga".

The ImageSelect Window



IMAGESELECTHEADER



WindowType (IconSli)

As with every window header the first button allows you to set the type of window

Full Window (IconTog)

Maximise the window or return to the previous window display size; return to the old screen setting.

HotKey: (ALT-)CTRL+UPARROW



Remove (IconBut)

Delete the ".Bpib" help file in the current directory. A new ".Bpib" is only created once the directory is read again.

Dirview (IconTog)

Indicates whether the left part, where the directories are displayed, is shown.

Info (IconTog)

Indicates whether the lower part, where information about the active picture is displayed, is shown.

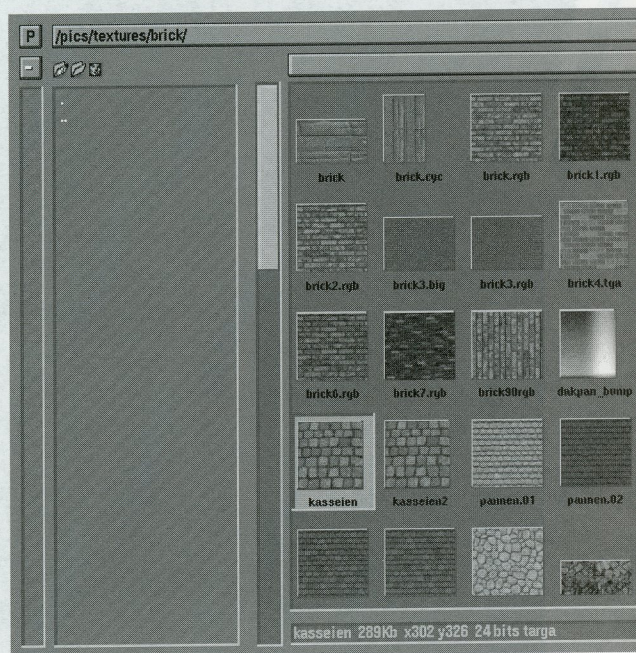
Images (IconTog)

Obsolete.

Magnify (IconTog)

The active picture is displayed twice as large.

IMAGESELECTWINDOW



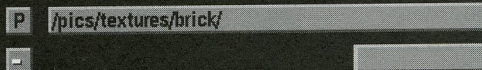
In parts of the Blender interface where pictures can be loaded, you generally have the option of using a FileSelect window or the ImageSelect window

For the most part, the functionality is the same.

The ImageSelect window reads the directory and examines each file to see if it is a recognisable image. After the entire directory

is scanned, the images are loaded and displayed as a thumbnail and saved in the ".Bpib" file.

If a ".Bpib" file already exists, it is first read in and compared to the contents of the directory.



P (But)

Displays the parent directory. Also with: PKEY.

DirName: (TextBut)

This button displays the current directory.

Preset Directories (MenuBut)

The file \$HOME/.Bfs contains a number of pre-sets that are shown in this menu.

While a file is being read or written, the directory involved is temporarily added to the menu.

FileName: (TextBut)

The file name can be entered here.



Status Icons.

The different phases of ImageSelect:

- Was a ".Bpib" file found?
- Was the directory scanned completely?
- Have all the pictures been read in?

THE MOUSE AND HOTKEYS

LeftMouse

Activate a file. The file name is placed in the FileName button.

MiddleMouse

Activate a file and return to the previous window type.

RightMouse

Select a file.

ENTER / PADENTER

Close the ImageSelectWindow; return with a OK message.

ESC

Close the ImageSelectWindow; no action is performed.

PAGEDN

Scroll down one page.

PAGEUP

Scroll up one page.

PKEY

Go to the parent directory.

The Animation Playback Window

To allow you to view sequential sequences of rendered frames, Blender has a simple, but efficient animation playback option. This is invoked with a button: `DisplayButtons→Play`. This button plays all of the numbered frames specified in the `DisplayButtons→pics` `TextBut`.

An alternative for starting the animation window is to type `-a` in the command line: `blender -a <first_image_file>`.

Blender first reads all the files in memory and then displays them as a flip book. Check in advance to make sure sufficient memory is available; you can see this with the `FileWindow`. Use `ESCKEY` to stop the reading process.

The commands available in the playback window are:

ESC

Close the window

ENTER, PADENTER

Start playback.

LEFTARROW, DOWNARROW

Stops the playback; if playback is already stopped, moves 1 frame back.

RIGHTARROW, UPARROW

Stops the playback; if playback is already stopped, moves 1 frame forward.

PAD_0

Sets the playback at the first frame and switches 'cyclical' playback off

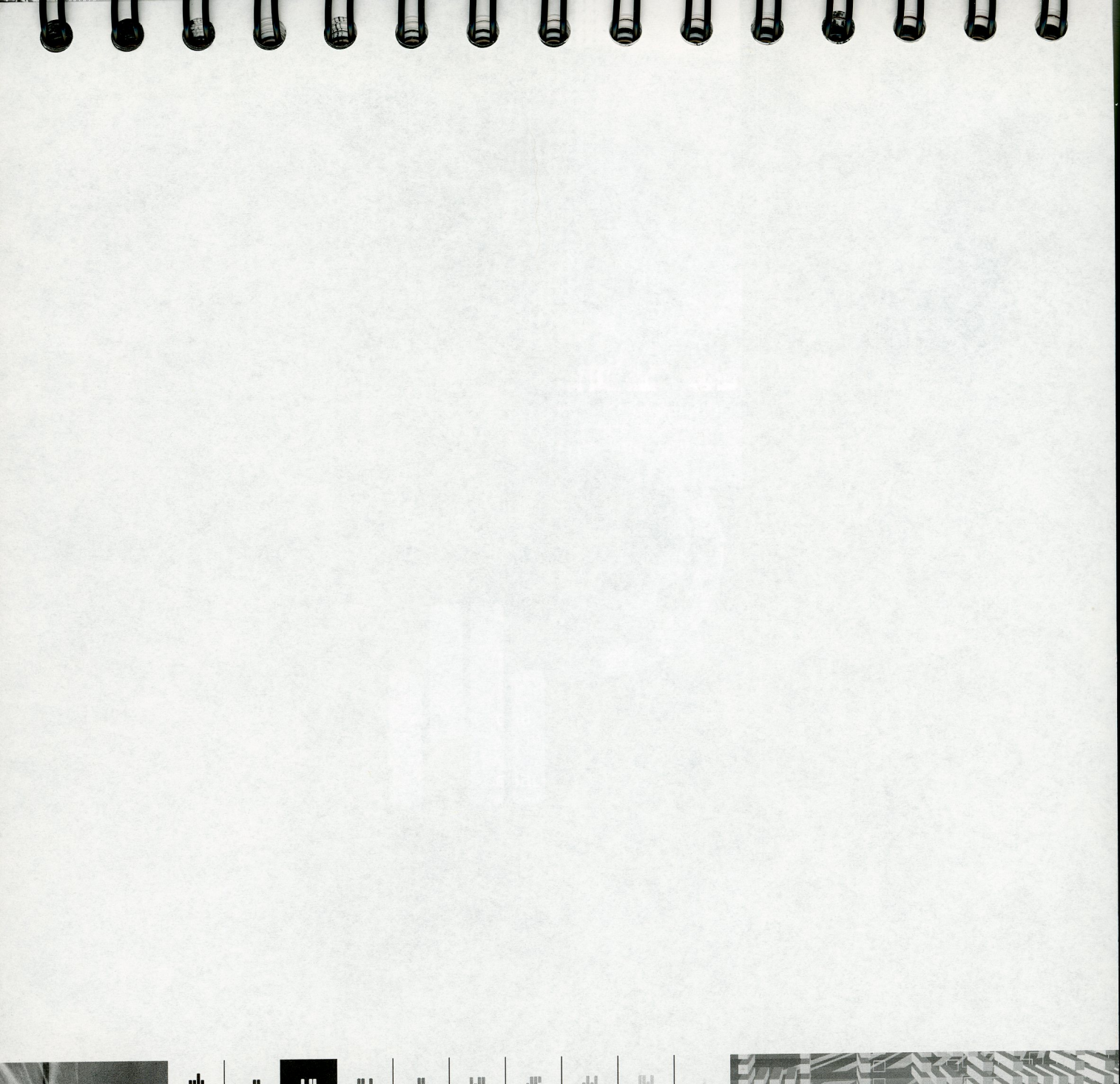
Pressing this key again turns cyclical playback on again and starts the playback at the beginning.

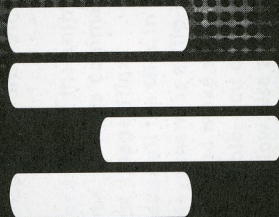
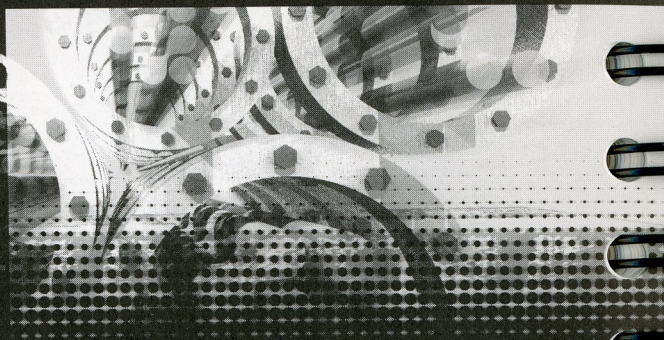
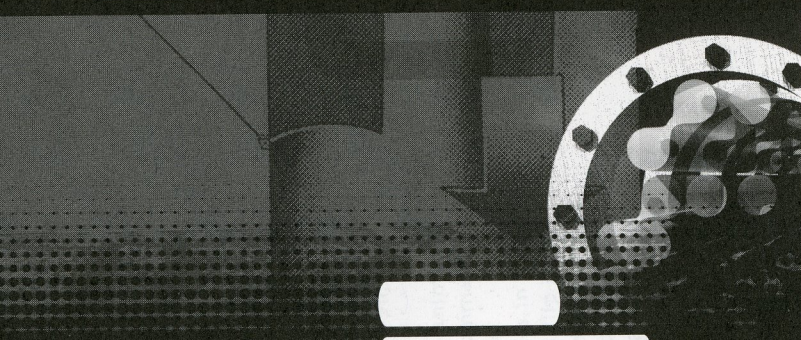
PAD_1 to PAD_9

The playback speed. 60, 50, 30, 25, 20, 15, 12, 10 and 6 frames per second, respectively.

LeftMouse (hold-move)

Move the mouse horizontally through the playback window to scroll through the frames.

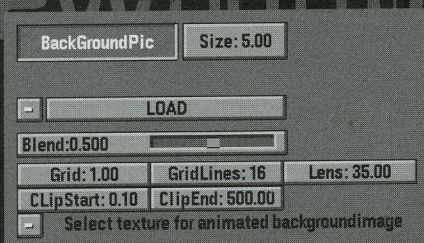




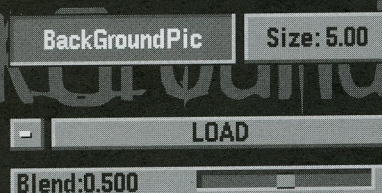
Part 9

Reference: the buttons

The ViewButtons



These buttons are not global; independent variables can be set for each 3Dwindow, for the *grid* or the *lens*, for example. Use the HotKey SHIFT+F7 in the 3DWindow or the WindowType button in the 3DHeader. Use SHIFT+F5 to restore the 3DWindow.



BackGroundPic (TogBut)

This option displays a picture in the background of the 3DWindow. Standard Image blocks are used; re-using an Image does not consume any additional memory. The BackGroundPic is only drawn in *ortho* and *Camera* view. It is always centered around the global nulpoint. In camera View, it is completely displayed in the viewport.

Size: (NumBut)

The size in Blender units for the width of the BackGroundPic. Only of interest in *ortho*.

ImageBrowse (MenuBut)

Select an existing Image from the list provided.

LOAD (But)

The window changes into an ImageSelectWindow. Use this to specify the picture to be used as the BackGroundPic. The picture is added to the Blender structure as an Image block.

Blend (NumSli)

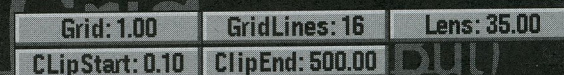
The factor with which the grey of the 3DWindow is blended with the background picture.

Select texture for animated backgroundimage

TextureBrowse (MenuBut)

Specify a Texture to be used as the BackGroundPic. This only works for Image Textures.

The TextureButtons have extensive options for an animated Image Texture, which also allows you to achieve an animated BackGroundPic. Use this option for *rotoscoping*, for example. This is a technique in which video images are used as examples or as a basis for 3D animation.



Grid: (NumBut)

The distance between two grid lines.

GridLines: (NumBut)

The number of lines that comprise the grid in *perspective* or *Camera* view.

Lens: (NumBut)

The 'lens' used for the *perspective* view. This is independent of the Camera.

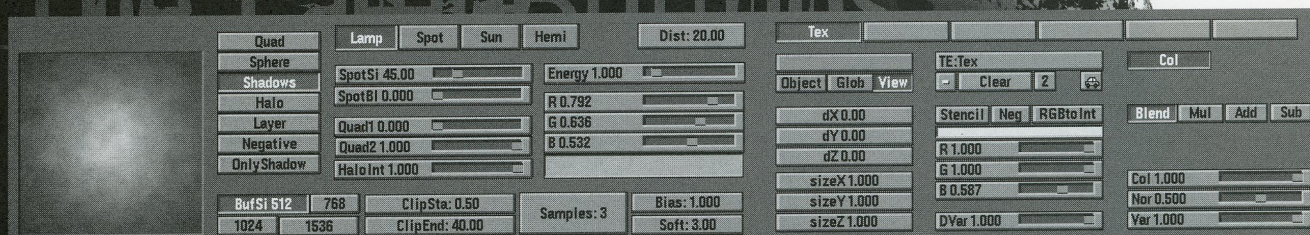
CLipStart: (NumBut)

The start clipping value in *perspective* view mode.

ClipEnd: (NumBut)

The distance beyond which items are no longer displayed in *perspective* view mode. The ClipStart and ClipEnd limits also determine the resolution (actually the density) of the OpenGL zbuffer. Try to keep these values as close together as possible, so that the zbuffer can distinguish small differences.

The LampButtons



The settings in these ButtonsWindow visualise the Lamp DataBlock. The LampButtons are only displayed if the active Object is a Lamp. HotKey for LampButtons: F4.

LA:Lamp 2

The DataButtons in the Header indicate what Lamp block is visualised.

Lamp Browse (MenuBut)

Choose another Lamp block from the list provided.

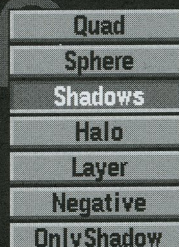
LA: (TextBut)

Give the current Lamp a new and unique name.

Users (But)

If the Lamp Block is used by more than one Object, this button shows the total number of Objects. Press the button to make the Lamp "Single User". Then an exact copy is made.

Lamp options.



Quad (TogBut)

The distance from the lamp is in inverse quadratic proportion to the intensity of the light. An inverse linear progression is standard (see also the buttons "Dist", "Quad1" and "Quad2").

Sphere (TogBut)

The lamp only sheds light within a spherical area around the lamp. The radius of the sphere is determined by the "Dist" button.

Shadows (TogBut)

The lamp can produce shadows. Shadow calculations are only possible with the Spot lamps. The render option "Shadows" must also be turned on in the DisplayButtons. See also the *shadow buffer* buttons later in this section.

Halo (TogBut)

The lamp has a halo. This only works with Spot lamps. The intensity of the halo is calculated using a conic section. It is correctly displayed if it is cut through by planes. A spot bundle shines 'through' planes. It does not use the

shadow algorithm. The scope of the spot halo is determined by the value of "Dist"

Layer (TogBut)

Only Objects in the same layer(s) as the Lamp Object are illuminated. This enables you to use selective lighting to give objects an extra accent or to restrict the effects of the lamp to a particular space. It also allows you keep rendering times under control.

Negative (TogBut)

A lamp casts 'negative' light.

OnlyShadow (TogBut)

For spot lamps (with "Shadow" ON), only the shadow is rendered. Light calculations are not performed and where there are shadows, the value of "Energy" is reduced.

Lamp types.



Lamp (RowBut)

The standard lamp, a point light source.

Spot (RowBut)

The lamp is restricted to a conical space. The 3DWindow shows the form of the spotlight with a broken line. Use the sliders "SpotSi" and "SpotBl" to set the angle and the intensity of the beam.

Sun (RowBut)

The light shines from a constant direction; the distance has no effect. The

position of the Lamp Object is thus unimportant, except for the rotation.

Hemi (RowBut)

Like "Sun" but now light is shed in the form of half a sphere, a *hemisphere*. This method is also called *directional ambient*. It can be used to suggest cloudy daylight.

Dist: 20.00

Dist (NumBut)

For the lamp types "Lamp" and "Spot" the distance affects the intensity of the light. The standard formula is used for this:

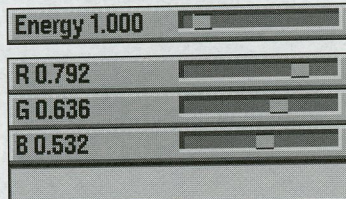
$D = \text{"Dist" button.}$

$a = \text{distance to the lamp.}$

$\text{Light intensity} = D/(D + a).$

This is an inverse linear progression. With the option "Quad" this becomes:

$\text{Light intensity} = D/(D + a \times a).$

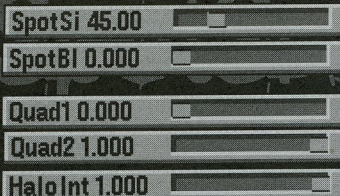


Energy (NumSlI)

The intensity of the light. The standard settings in Blender assume that a minimum of two lamps are used.

R, G, B (NumSlI)

The red, green and blue components of the light.



SpotSi (NumSli)

The angle of the beam measured in degrees. Use for shadow lamp beams of less than 160 degrees.

SpotBl (NumSli)

The softness of the spot edge.

Quad1, Quad2 (NumSli)

The light intensity formula of a Quad Lamp is actually: Light intensity =

$$D / (D + (\text{quad1} * a) + (\text{quad2} * a * a))$$

D = "Dist" button.

a = distance to the lamp.

The values of "quad1" and "quad2" at 1.0 produces the strongest quadratic progression. The values of "quad1" and "quad2" at 0.0 creates a special Quad lamp that is insensitive to distance.

HaloInt (NumSli)

The intensity of the spot halo. The scope of the spot halo is determined by "Dist".

Shadow Buffer

BufSi 512	768	ClipSta: 0.50	Samples: 3	Bias: 1.000
1024	1536	ClipEnd: 40.00		Soft: 3.00

Blender uses a *shadow buffer* algorithm. From the spotlight, a picture is rendered for which the distance from the spotlight is saved for each pixel. The shadow buffers are compressed, a buffer of 1024x1024 pixels requires, on average, only 1.5 Mb of memory. This method works quite quickly, but must be adjusted carefully. There are two possible side effects:

- *Aliasing*. The shadow edge has a block-like progression. Make the spot beam smaller, enlarge the buffer or increase the number of *samples* in the buffer
- *Biasing*. Faces that are in full light show *banding* with a block-like pattern. Set the "Bias" as high as possible and reduce the distance between "ClipSta" and "ClipEnd".

Bufsi 512, 768, 1024, 1536 (RowBut)

The size of the buffer in pixels. The value of DisplayButtons→Percentage (100%, 75%, ...) is multiplied by this.

ClipSta, ClipEnd (NumBut)

Seen from the spot lamp: everything closer than ClipSta always has light; everything farther away than ClipEnd always has shadow. Within these limits, shadows are calculated. The smaller the shadow area, the clearer the distinction the lamp buffer can make between small distances, and the fewer side effects you will have.

It is particularly important to set the value of ClipSta as high as possible.

Samples (NumBut)

The shadow buffer is 'sampled'; within a square area a test is made for shadow 3*3, 4*4 or 5*5 times. This reduces *aliasing*.

Soft (NumBut)

The size of the sample area. A large "Soft" value produces broader shadow edges.

Texture Channels



Texture name (RowBut)

A Lamp has six *channels* with which Textures can be linked. Each *channel* has its own *mapping*, i.e. the manner in which the texture

works on the lamp. The settings are in the buttons described below.

Mapping: coordinates input.



Each Texture has a 3D coordinate (the texture coordinate) as input. The starting point is always the global coordinate of the 3D point that is seen in the pixel to be rendered.

A lamp has three options for this.

Object Name (TextBut)

The name of the Object that is used for the texture coordinates. If the Object does not exist, the button remains empty.

Object (RowBut)

Each Object in Blender can be used as a source for texture coordinates. To do this, an inverse transformation is applied to the global coordinate, which gives the *local* Object coordinate. In this way, the texture is linked with the position, size and rotation of the Object.

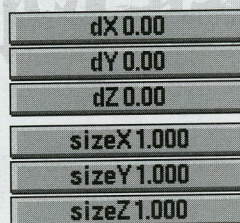
Glob (RowBut)

The global coordinate is passed on to the texture.

View (RowBut)

The view vector of the lamp; the vector of the global coordinate to the lamp, is passed on to the texture. If the lamp is a Spot, the view vector is normalised to the dimension of the spot beam, allowing use of a Spot to project a 'slide'

Mapping: transform coordinates.



Use these buttons to adjust the texture coordinate more finely.

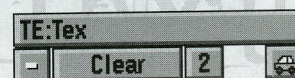
dX, dY, dZ (NumBut)

The extra translation of the texture coordinate.

sizeX, sizeY, sizeZ (NumBut)

The extra scaling of the texture coordinate.

Texture Block



TE: (TextBut)

The name of the Texture block. The name can be changed with this button.

Texture Browse (MenuBut)

Select an existing Texture from the list provided, or create a new Texture Block.

Clear (But)

The link to the Texture is erased.

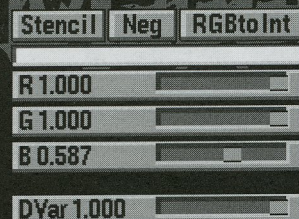
Users (But)

If the Texture Block has multiple users, this button shows the total number of users. Press the button to make the Texture "Single User" Then an exact copy is made.

Auto Name (But)

Blender assigns a name to the Texture.

Mapping: Texture input settings.



These buttons pass extra information to the Texture.

Stencil (TogBut)

Normally, textures are executed one after the other and placed over each other. A second Texture *channel* can completely replace the first.

This option sets the *mapping* to *stencil* mode. No subsequent Texture can have an effect on the area the current Texture affects.

Neg (TogBut)

The inverse of the Texture is applied.

RGBtoInt (TogBut)

With this option, an RGB texture (affects colour) is used as an Intensity texture (affects a value).

R, G, B (NumSli)

The colour with which an Intensity texture blends with the current colour.

DVar (NumSli)

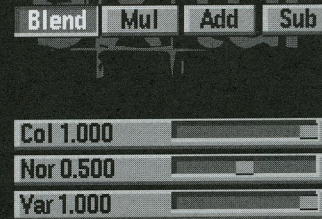
The value with which the Intensity texture blends with the current value.

Mapping: output to.



The texture affects the colour of the lamp.

Mapping: output settings.



These buttons adjust the output of the Texture.

Blend (RowBut)

The Texture mixes the values.

Mul (Rowbut)

The Texture multiplies the values.

Add (RowBut)

The Texture adds the values.

Sub (RowBut)

The Texture subtracts the values.

Col (NumSli)

The extent to which the texture affects the colour.

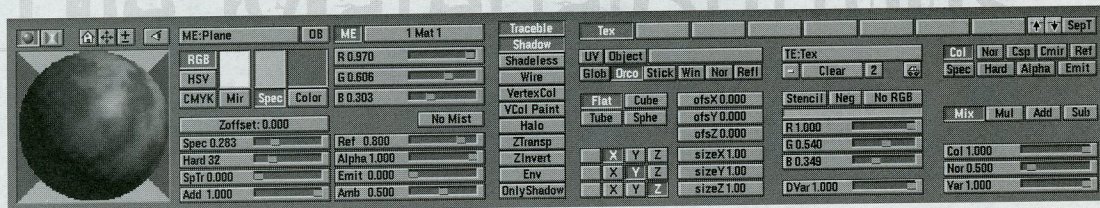
Nor (NumSli)

The extent to which the texture affects the normal (not important here).

Var (NumSli)

The extent to which the texture affects the value (a variable, not active here).

The MaterialButtons



The settings in this ButtonsWindow visualise the Material DataBlock. The MaterialButtons are only displayed if the active Object has a Material. Hotkey: F5.



The DataButtons in the Header indicate what Material block is visualised.

Material Browse (MenuBut)

Select another Material from the list provided, or create a new block.

MA: (TextBut)

Give the current Material a new and unique name.

Users (But)

If the Material block is used by more than one Object, this button indicates the total number of users. Press the button to make the Material "Single User". An exact copy is created.

Remove Link (But)

Delete the link to the Material.

Auto Name (But)

Blender assigns a name to the Material



Copy to buffer (IconBut)

The complete contents of the Material and all the mapping is copied to a temporary buffer

Copy from buffer (IconBut)

The temporary buffer is copied to the Material.

Preview settings.



Only the first two buttons are active in this version.

Sphere (IconTog)

There are two built-in preview options. The preview plane shows the X-Y coordinates, the Z axis is the vertical axis for the preview sphere; the X and Y axes revolve around this axis.

Background (IconTog)

Use this button to select a light or a dark background.



These buttons specify what the Material block is linked to, or must be linked to.

By linking Materials directly to Objects, each Object is rendered in its own Material.

ME: (TextBut)

This Button specifies the block to which the Material is linked. This button can only be used to give the block another name.

Possible blocks are:

- ME: Material is linked to a Mesh (ObData) block.
- CU: Material is linked to a Curve, Surface or Font (ObData) block.
- MB: Material is linked to a MetaBall (ObData) block.
- OB: Material is linked to the Object itself.

OB (RowBut)

Use this button to link the current Material to the *Object*. Any link to the ObData block remains in effect. Links can be removed with the Header button: "Remove Link".

ME or CU or MB (RowBut)

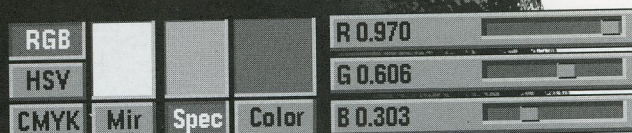
Use this button to be able to link a Material to the *ObData* of the Object. Any link to the Object block remains in effect. Links can be removed with the Header button: "Remove Link".

1 Mat 1

1 Mat 1 (NumBut)

An Object or ObData block may have more than one Material. This button can be used to specify which of the Materials must be displayed, i.e. which Material is *active*. The first digit indicates how many Materials there are; the second digit indicates the number of the *active* Material.

Each face in a Mesh has a corresponding number: the 'Material index'. The number of *indices* can be specified with the EditButtons. Curves and Surfaces also have Material indices.



RGB (RowBut)

Most colour sliders in Blender have two pre-set options: in this case, the colour is created by mixing Red, Green, Blue.

HSV (RowBut)

The colour sliders mix colour with the Hue, Saturation, Value system. 'Hue' determines the colour, 'Saturation' determines the amount of colour in relation to grey and 'Value' determines the light intensity of the colour.

CMYK (RowBut)

Obsolete.

The following buttons specify what type of color is visualised in the sliders:

Mir (RowBut)

The mirror colour of the Material (Obsolete).

Spec (RowBut)

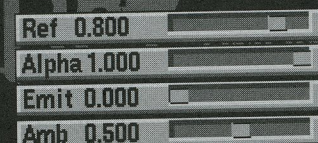
Specularity, the colour of the sheen.

Color (RowBut)

The basic colour of the Material.

R, G, B (NumSli) or H, S, V (NumSli)

These mix the colour specified.



Ref (NumSli)

Reflectivity. The degree to which the Material reflects the basic colour when light falls on it.

Materials

Alpha (NumSli)

The degree of coverage, which can be used to make Materials transparent. Use the option “ZTransp” to specify that multiple transparent layers can exist. Without this option, only the Material itself is rendered, no matter what faces lie behind it.

The transparent information is saved in an *alpha layer* which can be saved as part of a picture (see DisplayButtons).

Emit (NumSli)

The Material ‘emits light’ without shedding light on other planes of course.

Ambient (NumSli)

The degree to which the global Ambient colour is applied, a simple form of environmental light. The global Ambient can be specified in the World block, using the WorldButtons. Ambient is useful for giving the total rendering a softer more coloured atmosphere.

Zoffset: 0.000	
Spec 0.283	☐
Hard 32	☐
SpTr 0.000	☐
Add 1.000	☐

Zoffset (NumBut)

This button allows you to give the faces to be rendered an artificial forward offset in Blender’s Zbuffer system. This only applies to Materials with the option “ZTransp”

This option is used to place cartoon figures on a 3D floor as images with alpha. To prevent the figures from ‘floating’ the feet and the

shadows drawn must be placed partially beneath the floor. The Zoffset option then ensures that the entire figure is displayed. This system offers numerous other applications for giving (flat) images of spatial objects the appropriate 3D placement.

Spec (NumBut)

The degree of sheen (specularity) the material has.

Hard (NumBut)

The hardness of the specularity. A large value gives a hard, concentrated sheen, like that of a billiard ball. A low value gives a metallic sheen.

SpTr (NumBut)

This button makes areas of the Material with a sheen opaque. It can be used to give transparent Materials a ‘glass’ effect.

Add (NumBut)

Obsolete.

Traceable
Shadow
Shadeless
Wire
VertexCol
VCol Paint
Halo
ZTransp
ZInvert
Env
OnlyShadow
No Mist

Traceable (TogBut)

This term stems from Blender’s ray-trace past. It specifies whether or not shadow lamps can ‘see’ the current Material. Turn the “Traceable” option OFF to prevent undesired shadows.

Shadow (TogBut)

This button determines whether the Material can receive a shadow, i.e. whether a shadow calculation is needed.

Shadeless (TogBut)

This button makes the Material insensitive to light or shadow

Wire (TogBut)

Only the *edges* of faces are rendered (normal rendering!). This results in an exterior that resembles a wire frame. This option can only be used for Meshes.

VertexCol (TogBut)

If the Mesh vertex has colours (see EditButtons), they are added to the Material as extra *light*. The colours also remain visible without lamps. Use this option to render *radiosity-like* models.

VertexPaint (TogBut)

If the Mesh vertex has colours, this button replaces the basic colour of the Material with these colours. Now light must shine on the Material before you can see it.

Halo (TogBut)

Instead of rendering the faces, each vertex is rendered as a halo. The *lens flare* effect is a part of the halo.

This option change certain MaterialButtons (see the following section).

ZTransp (TogBut)

Transitional Zbuffers can only render opaque faces. Blender uses a modified method to Zbuffer transparent faces. This method requires more memory and calculation time than the normal Zbuffer, which is why the two systems are used alongside each other

Zinvert (TogBut)

The Material is rendered with an inverse Zbuffer method; front and back are switched.

Env (TogBut)

Environment option. The Material is not rendered and the Zbuffer and render buffers are 'erased' so that the pixel is delivered with Alpha = 0.0.

OnlyShadow (TogBut)

This option determines the *alpha* for transparent Materials based on the degree of shadow. Without a shadow the Material is not visible.

NoMist (TogBut)

The Material is insensitive to "Mist" (see WorldButtons).

Texture Channels.



Texture name (RowBut)

A Material has eight *channels* to which Textures can be linked. Each *channel* has its own *mapping*, which is the effect the texture has on the material.



Copy to buffer (IconBut)

The complete *mapping* settings are copied to a temporary buffer.

Copy from buffer (IconBut)

The contents of the temporary buffer are copied to the *mapping* settings.

SepT (TogBut)

Separate Textures. This option forces only the current *channel* to be rendered with its corresponding Texture.

Mapping: coordinates as input.



Each Texture has a 3D coordinate (the texture coordinate) as input. The starting point is generally the global coordinate of the 3D point that can be seen in the pixel to be rendered. A Material has the following Mapping options:

UV (RowBut)

The U-V coordinates of a face or Nurbs surface from an Object make up the texture coordinates. U-V is a commonly used term for specifying the mathematical space of a flat or curved surface.

Object (RowBut)

Every Object in Blender can be used as a source for texture coordinates. For this, the Object's inverse transformation is applied to the global coordinate, to obtain *local* Object coordinates. This links the texture to the position, dimension and rotation of the Object.

Generally an Empty Object is used to specify the exact location of a Texture, e.g. to place a logo on the body of an airplane. Another commonly used approach is to have the 'Texture Object' move to achieve an animated texture.

Object Name (TextBut)

The name of the Object used for the texture coordinates. If the Object does not exist, the button remains empty

Glob (RowBut)

The global coordinate is passed on to the texture.

Orco (RowBut)

The standard setting. This is the *original coordinate* of the Mesh or another ObData block.

Stick (RowBut)

Sticky texture. Blender allows you to assign a texture coordinate to Meshes, which is derived from the manner in which the Camera view sees the Mesh. The screen coordinate (only X,Y) for

each vertex is calculated and saved in the Mesh. This makes it appear as if the texture is projected from the Camera; the texture becomes "sticky" (see also "Make Sticky" in the EditButtons).

Use "Sticky" to precisely match a 3D object with an Image Texture. Special *morphing* effects can also be achieved.

Win (RowBut)

The screen coordinate (X,Y) is permanently used as a texture coordinate. Use this method to achieve 2D *layering* of different Images.

Nor (RowBut)

The normal vector of the rendered face is used as a texture coordinate. Use this method to achieve *reflection mapping*, which is the suggestion of mirroring using a specially pre-calculated Image.

Refl (RowBut)

The reflection vector of the rendered face is used as a texture coordinate. This vector points in a direction that makes the face appear to be mirrored. Use this option to suggest a reflected surface with procedural textures such as "Marble" or "Clouds"

Mapping: transform coordinates.

ofsX 0.000

ofsY 0.000

ofsZ 0.000

Use these buttons to more finely adjust the texture coordinate.

sizeY 1.00

sizeZ 1.00

dX, dY, dZ (NumBut)
The extra translation of the texture coordinate.

sizeX, sizeY, sizeZ (NumBut)

The extra scaling of the texture coordinate.

Mapping: 3D to 2D



For Image Textures only, this determines the manner in which the 3D coordinate is converted to 2D.

Flat (RowBut)

The X and Y coordinates are used directly.

Cube (RowBut)

Depending on the normal vector of the plane, the X-Y or the X-Z or the Y-Z coordinates are selected. This option works well for stones, marbles and other regular textures,

Tube (RowBut)

This creates a tube-shaped mapping. The Z axis becomes the central axis, X and Y revolve around it.

Sphere (RowBut)

This causes a sphere-shaped mapping. The Z axis becomes the central axis, X and Y revolve around it.

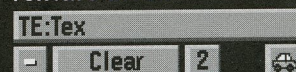
Mapping: switch coordinates.



The three rows of buttons indicate the new X, Y and Z coordinates. Normally, the X is mapped to X, the Y to Y and Z to Z.

The first button switches a coordinate completely off.

Texture Block



TE: (TextBut)

The name of the Texture block. The name can be changed with this button.

Texture Browse (MenuBut)

Select an existing Texture from the list provided, or create a new Texture Block.

Clear (But)

The link to the Texture is erased.

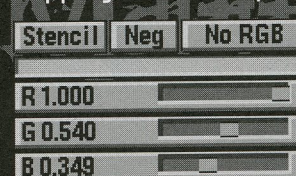
Users (But)

If the Texture Block has multiple users, this button displays the total number of users. Press the button to make the Texture "Single User". An exact copy is made.

Auto Name (But)

Blender assigns a name to the Texture.

Mapping: Texture input settings.



These buttons pass extra information to the Texture.

Stencil (TogBut)

Normally, textures are executed one after the other and laid over one another.

A second Texture *channel* can completely replace the first.

With this option, the *mapping* goes into *stencil* mode. No subsequent Texture can have an effect on the area the current Texture affects.

Neg (TogBut)

The effect of the Texture is reversed.

RGBtoInt (TogBut)

With this option, an RGB texture (affects colour) is used as an Intensity texture (affects a value).

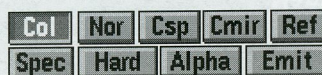
R, G, B (NumSli)

The colour with which an Intensity texture blends with the current colour

DVar (NumSli)

The value with which the Intensity texture blends with the current value.

Mapping: output to.



Col (TogBut)

The texture affects the basic colour of the material.

Nor (TogBut)

The texture affects the rendered normal. Only important for Image textures. The "Stucci" texture does this itself

Csp (TogBut)

The texture affects the *specularity* colour of the material.

Cmir (TogBut)

The texture affects the *mirror* colour of the material.
(Obsolete)

Ref (Tog3But)

The texture affects the value of the material's

reflectivity. There are three settings; the third setting reverses the effect

Spec (Tog3But)

The texture affects the value of *specularity* of the material. There are three settings.

Hard (Tog3But)

The texture affects the *hardness* value of the material. There are three settings.

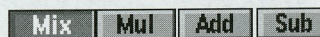
Alpha (Tog3But)

The texture affects the *alpha* value of the material. There are three settings.

Emit (Tog3But)

The texture affects the "Emit" value of the material. There are settings.

Mapping: output settings.



These buttons change the output of the Texture.

Mix (RowBut)

The Texture blends the values or colour

Mul (Rowbut)

The Texture multiplies values or colour

Add (RowBut)

The Texture adds the values or colour

Sub (RowBut)

The Texture subtracts values or colour

Col (NumSli)

The extent to which the texture affects colour





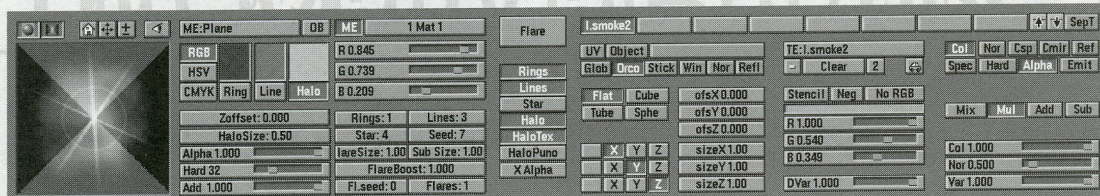
Nor (NumS1i)

The extent to which the texture affects the normal (not important here).

Var (NumS1i)

The extent to which the texture affects a value (a variable).

The MaterialButtons, Halos



If a Material has the option "Halo" ON, a number of buttons change to specific halo settings. *Lens flares* can also be created here.

Halos are rendered on the 3D location of the vertices. These are small, transparent round spots or pictures over which circles and lines can be drawn.

They take Blender's Zbuffer into account; like any 3D element, they can simply disappear behind a face in the forefront.

Halos are placed over the currently rendered background as a separate layer or they give information to the *alpha layer* allowing halos to be processed as a post-process.

Only Meshes and Particle Effects can have halos. A Mesh with a halo is displayed differently in the 3DWindow; with small dots at the position of the vertices.

Halos cannot be combined with 'ordinary' faces within one Mesh. Only one Material can be used per 'halo' Mesh.

Flare

Rings

Lines

Star

Halo

HaloTex

HaloPuno

X Alpha

Flare (TogBut)

Each halo is now also rendered as a *lens flare*. This effect suggests the reflections that occur in a camera lens if a strong light source shines on it. A Flare consists of three layers: the ordinary halo, which has a 3D location, and can thus disappear behind a face. the basic Flare, which is the same halo, but possibly with other dimensions. This is placed over the entire

rendering as a post-process.

the sub Flares, multi-coloured dots and circles, that are also placed over the entire rendering as a post-process.

The "HaloSize" value not only determines the dimensions, but is also used to determine the visibility and thus the strength of the Flare rendered in the post-process. This way a Flare that disappears slowly behind a faces will decrease in size at a corresponding speed and gradually go out.

Rings (TogBut)

Determines whether rings are rendered over the basic halo.

Lines (TogBut)

Determines whether star-shaped lines are rendered over the basic halo.

Star (TogBut)

Instead of being rendered as a circle, the basic halo is rendered in the shape of a star. The NumBut "Star" determines the number of points the star has.

Halo (TogBut)

Turn this option OFF to return to a normal Material.

HaloTex (TogBut)

Halos can be given textures in two ways:

- "HaloTex" OFF: the basic colour of each halo is determined by the texture coordinate of the halo-vertex.
- "HaloTex" ON: each halo gets a complete texture area, in which, for example, an Image texture is displayed completely in each basic halo rendered.

HaloPuno (TogBut)

The vertex normal ("Puno" in Blender's turbo language) is used to help specify the dimension of the halo. Normals that point directly at the Camera are the largest; halos with a normal that point to the rear are not rendered.

If there are no vertex normals in the Mesh (the Mesh only consists of vertices) the normalised local coordinate of the vertex is used as the normal.

XAlpha

Extreme Alpha. Halos can 'emit light'; they can add colour. This cannot be expressed with a normal *alpha*. Use this option to force a stronger progression in the alpha.



Ring (RowBut)

With this option ON, the colour of the rings can be mixed with the RGB sliders.

Line (RowBut)

With this option ON, the colour of the lines can be mixed with the RGB sliders.

Halo (RowBut)

With this option ON, the colour of the basic halo can be mixed with the RGB sliders.

R, G, B (NumSli)

Use these sliders to mix the indicated colour.



HaloSize (NumBut)

The dimension of the halo.

Alpha (NumSli)

The degree of coverage of the halo.

Hard (NumSli)

The hardness of the halo, a large value gives a strong, concentrated progression.

Add (NumSli)

Normally, the colour of halos is calculated during rendering, giving a light emitting effect. Set the "Add" value to 0.0 to switch this off and make black or 'solid' halos possible as well.

Flare Buttons

Rings: 1	Lines: 3
Star: 4	Seed: 7
FlareSize: 1.00	Sub Size: 1.00
FlareBoost: 1.000	
Fl.seed: 0	Flares: 1

Rings (NumBut)

The number of rings rendered over the basic halo.

Lines (NumBut)

The number of star-shaped lines rendered over the basic halo.

Star (NumBut)

The number of points on the star-shaped basic halo.

Seed (NumBut)

'Random' values are selected for the dimension of the *rings* and the location of the *lines* based on a fixed table. "Seed" determines an offset in the table.

FlareSize (NumBut)

The factor by which the post-process basic Flare is larger than the halo.

SubSize (NumBut)

The dimension of post-process sub Flares, multicolored dots and circles.

FlareBoost (NumBut)

This gives the Flare extra strength.

Fl.seed (NumBut)

The dimension and shape of the sub Flares is determined by a fixed table with 'random' values. "Fl.seed" specifies an offset in the table.

Flares (NumBut)

The number of sub Flares.

The TextureButtons

The settings in this ButtonsWindow visualise the Texture DataBlock. These buttons are only displayed if:

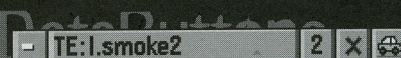
- the active Material has a Texture (see MaterialButtons).
- the active Lamp has a Texture (see LampButtons).
- the World block has a Texture (see WorldButtons)

Blender automatically selects the correct setting if the TextureButtons are called up from the MaterialButtons, LampButtons or WorldButtons.

Hotkey: F6.

Each Texture has a 3D coordinate (the texture coordinate) as input. What happens here is determined by the type of texture:

- Intensity textures: return one value. The *preview render* in this window shows this as grey values.
RGB textures: returns three values; they always work on colour.
- Bump textures: returns three values; they always work on the normal vector. Only the "Stucci" and "Image" texture can give normals.



The DataButtons in the Header indicate what Texture block is visualised.

Texture Browse (MenuBut)

Select another Texture from the list provided, or create a new block.

TE: (TextBut)

Give the current Texture block a new and unique name.

Users (But)

If the Texture block has more than one user, this button shows the total. Press the button to make the Texture "Single User". An exact copy is then created.

Remove Link (But)

Delete the link to the Texture.

Auto Name (But)

Blender assigns a name to the Texture.

THE STANDARD TEXTUREBUTTONS

MA: Material

Mat

World

Lamp

This group of buttons determines the type of user from which the Texture must be displayed.

Blender automatically selects the correct settings if the TextureButtons are invoked from the MaterialButtons, LampButtons or WorldButtons.

MA (or LA: or W0:) (TextBut)

The name of the DataBlock that has a (possible) link to the Texture.

Mat (RowBut)

The Texture of the active Material is displayed.

World (RowBut)

The Texture of the World block is displayed.

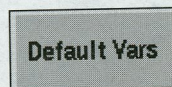
Lamp (RowBut)

The Texture of the Lamp is displayed.



This group of buttons shows the *channels*. In this example, we see that of the eight available *channels* for the Material only the first is linked to a Texture.

The program includes nine types of textures (see image at the bottom of the page). These are described in detail later in this manual.



Default Vars (But)

Changes the values set for the type of texture to the standard values.



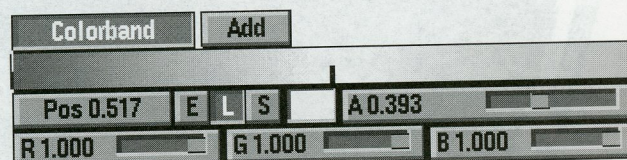
Bright (NumSli)

The 'brightness' of the colour or intensity of a texture. In fact, a fixed number is added or subtracted.

Contr (NumSli)

The 'contrast' of the colour or intensity of a texture. This is actually a multiplication.

COLORBAND



Use this option to create a smooth colour progression. Intensity textures are thus changed into an RGB texture.

The use of Colorband with a sharp transistion can cause *aliasing*.

Colorband (TogBut)

Switches the use of Colorband on or off

Add (TogBut)

Adds a new colour to the Colorband.

Cur: (NumBut)

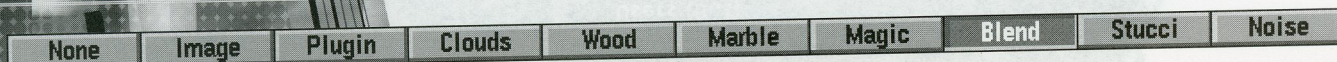
The active colour from the Colorband.

Del (TogBut)

Delete the active colour

Pos: (NumBut)

The position of the active colour Values range from 0.0 to 1.0. This can also be entered using LeftMouse (hold-move) in the Colorband.



InterPol	UseAlpha	CalcAlpha	NegAlpha	MipMap	Fields	Rot90	Movie	Anti	St. Field	Sharper
----------	----------	-----------	----------	--------	--------	-------	-------	------	-----------	---------

E, L, S (RowBut)

The interpolation type with which colours are mixed, i.e. 'Ease' 'Linear' and 'Spline'. The last gives the most fluid progression.

A, R, G, B (NumSli)

The Alpha and RGB value of the active colour.

InterPol (TogBut)

This option interpolates the pixels of an Image. This becomes visible when you enlarge the picture.

Turn this option OFF to keep the pixels visible. They are correctly anti-aliased. This last feature is useful for regular patterns, such as lines and tiles; they remain 'sharp' even when enlarged considerably.

UseAlpha (TogBut)

Use the *alpha* layer of the Image.

CalcAlpha (TogBut)

Calculate an *alpha* based on the RGB values of the Image.

NegAlpha (TogBut)

Reverses the *alpha* value.

MipMap (TogBut)

Generates a series of pictures, each half the size of the former one. This optimises the filtering process. When this option is OFF, you generally get a sharper image, but this can significantly increase calculation time if the filter dimension becomes large.

Fields (TogBut)

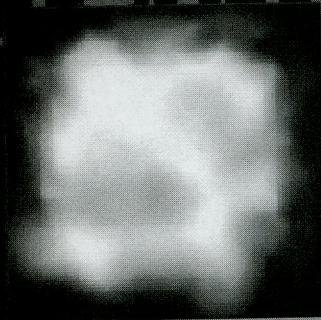
Video frames consist of two different images (fields) that are merged by horizontal line. This option makes it possible to work with field images. It ensures that when 'Fields' are rendered (DisplayButtons→Field) the correct field of the Image is used in the correct field of the rendering.

MipMapping cannot be combined with "Fields".

Rot90 (TogBut)

Rotates the Image 90 degrees when rendered.

IMAGE TEXTURE



The Image texture is the most frequently used and most advanced of Blender's textures. The standard bump-mapping and perspective-corrected MipMapping, filtering and anti-aliasing built into the program guarantee outstanding image quality (set the DisplayButtons→OSA ON for this). Because pictures are two-dimensional, you must specify in the *mapping* buttons how the 3D texture coordinate is converted to 2D; *mapping* is a part of the MaterialButtons.

Load Image

/pics/textures/halo/smoke2

1

Reload

Movie (TogBut)

SGI movies (only the SGI version of Blender) and 'anim5' files can also be used for an Image. To do this, set the "Frames" NumBut to the total number of frames.

Anti (TogBut)

Graphic images such as cartoons and pictures that consist of only a few colours with a large surface filling can be *anti-aliased* as a built in pre-process.

St Field (TogBut)

Normally the first field in a video frame begins on the first line. Some *frame grabbers* do this differently!

Filter : 1.00

Filter (NumBut)

The filter size used by the options "MipMap" and "Interpol"

(see image at the top of this page)

Load Image (But)

The (largest) adjacent window becomes an ImageSelectWindow. Specify here what file must be read to become an Image.

(But)

This small button does the same thing but now simply gives a FileSelect.

ImageBrowse (MenuBut)

You can select a previously created Image from the list provided. Image blocks can be reused without taking up extra memory.

File Name (TextBut)

Enter a file name here, after which a new Image block is created.

Users (But)

Indicates the number of users for the Image. The "Single User" option cannot be activated here. It has no significance for Images.

Reload (But)

Force the Image file to be read again.

Extend

Clip

ClipCube

Repeat

Xrepeat: 1

Yrepeat: 1

MinX 0.000

MinY 0.000

MaxX 1.000

MaxY 1.000

The following options determine what happens if the texture coordinate falls outside the Image.

Extend (RowBut)

Outside the Image the colour of the edge is extended.

Clip (RowBut)

Outside the Image, an *alpha* value of 0.0 is returned. This allows you to 'paste' a small logo on a large object.

ClipCube (RowBut)

The same as "Clip" but now the 'Z' coordinate is calculated as well. Outside a cube-shaped area around the Image, an *alpha* value of 0.0 is returned.

Repeat (RowBut)

The Image is repeated horizontally and vertically.

Xrepeat (NumBut)

The (extra) degree of repetition in the X direction.

Yrepeat (NumBut)

The (extra) degree of repetition in the Y direction.

MinX, MinY, MaxX, MaxY (NumBut)

Use these to specify a *cropping*, it appears that the Image actually becomes larger or smaller

Frames : 0	
Offset : 0	
Fie/Ima: 2	Cyclic
StartFr: 1	
Len: 0	

Frames (NumBut)

This activates the animation option; another image file (in the same Image block) will be read per rendered frame. Blender tries to find the other files by changing a number in the file name. Only the rightmost digit is interpreted for this. For example:

01.ima.099.tga + 1 becomes 01.ima.100.tga.

The value of "Frames" indicates the total number of files to be used.

If the option "Movie" is on, this value must also be set. Now, however, a frame is continually taken from the same file.

Offset (NumBut)

The number of the first picture of the animation.

Fie/Ima (NumBut)

The number of *fields* per rendered frame. If no *fields* are rendered, even numbers must be entered here. (2 fields = 1 frame).

Cyclic (TogBut)

The animation Image is repeated cyclically.

StartFr: (NumBut)

The moment - in Blender frames - at which the animation Image must start.

Len (NumBut)

This button determines the length of the animation. By assigning "Len" a higher value than "Frames", you can create a *still* at the end of the animation.

Fra: 0	0
Fra: 0	0
Fra: 0	0
Fra: 0	0

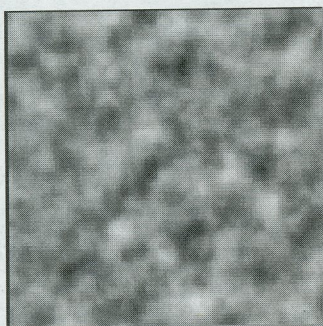
These buttons allow you to create a simple montage within an animation Image.

The left button, "Fra" indicates the frame number, the right-hand button indicates how long the frame must be displayed.

PLUG-IN TEXTURE

Texture plug-ins are not supported in this Blender version.

CLOUDS TEXTURE



"Clouds" is a *procedural texture*. This means that each 3D coordinate can be translated directly to a colour or a value. In this case, a three-dimensional table with pseudo random values is used, from which a fluent interpolation value can be calculated with each 3D coordinate (thanks to Ken Perlin for his masterful article "An Image Synthesizer" from the SIGGRAPH proceedings 1985). This calculation method is also called (*Perlin*) Noise.

Default

Color

Default (RowBut)

The standard Noise, gives an Intensity.

Color (RowBut)

The Noise gives an RGB value.

NoiseSize : 0.250

NoiseDepth: 2

NoiseSize (NumBut)

The dimension of the Noise table.

NoiseDepth (NumBut)

The depth of the Cloud calculation. A higher number results in a long calculation time, but also in finer details.

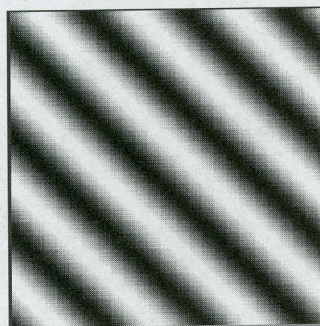
Soft noise

Hard noise

Soft Noise, Hard Noise (RowBut)

There are two methods available for the Noise function.

WOOD TEXTURE



"Wood" is also a *procedural texture*. In this case, bands are generated based on a sine formula. You can also add a degree of turbulence with the Noise formula. It returns an Intensity value only

Bands

Rings

BandNoise

RingNoise

Bands (RowBut)

The standard Wood texture.

Rings (RowBut)

This suggests 'wood rings'.

BandNoise (RowBut)

Applying Noise gives the standard Wood texture a certain degree of turbulence.

RingNoise (RowBut)

Applying Noise gives the rings a certain degree of turbulence.

NoiseSize : 0.250

Turbulence: 5.00

NoiseSize (NumBut)

The dimension of the Noise table.

Turbulence (NumBut)

The turbulence of the "BandNoise" and "RingNoise" types.

Soft noise

Hard noise

Soft Noise, Hard Noise (RowBut)

There are two methods available for the Noise function.

Soft

Sharp

Sharper

Soft, Sharp, Sharper (RowBut)

Three pre-sets for soft to more clearly defined Marble.

NoiseSize : 0.250

NoiseDepth: 3

Turbulence: 7.30

NoiseSize (NumBut)

The dimensions of the Noise table.

NoiseDepth (NumBut)

The depth of the Marble calculation. A higher value results in greater calculation time, but also in finer details.

Turbulence (NumBut)

The turbulence of the sine bands.

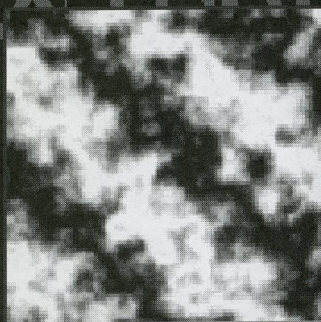
Soft noise

Hard noise

Soft Noise, Hard Noise (RowBut)

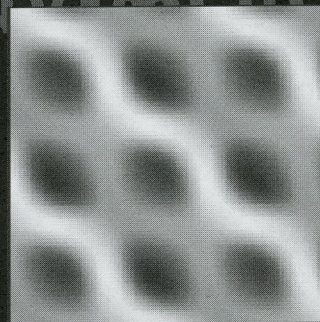
The Noise function works with two methods.

MARBLE TEXTURE



"Marble" is also a *procedural texture*. In this case, bands are generated based on a sine formula and Noise turbulence. It returns an Intensity value only.

MAGIC TEXTURE



"Magic" is a procedural texture. The RGB components are generated independently with a sine formula.

Size : 0.250

Turbulence: 5.00

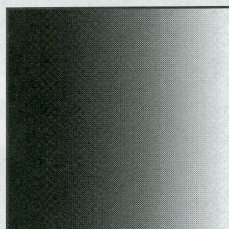
Size (NumBut)

The dimensions of the pattern.

Turbulence (NumBut)

The strength of the pattern.

BLEND TEXTURE



This is also a *procedural texture*. It generates a progression in Intensity.

Lin

Quad

Ease

Diag

Sphere

Halo

Lin (RowBut)

A linear progression.

Quad (RowBut)

A quadratic progression.

Ease (RowBut)

A flowing non-linear progression.

Diag (RowBut)

A diagonal progression.

Sphere (RowBut)

A progression with the shape of a three-dimensional ball.

Halo (RowBut)

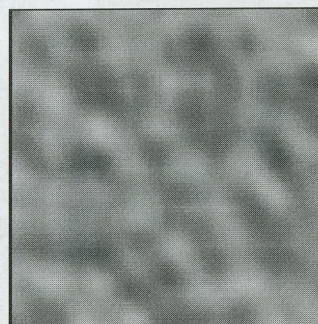
A quadratic progression with the shape of a three-dimensional ball.

Flip XY

Flip XY

The direction of the progression is flipped a quarter turn.

STUCCI TEXTURE



This *procedural texture* generates Noise-based normals.

Plastic

Wall In

Wall Out

Plastic (RowBut)

The standard Stucci

Wall In, Wall out (RowBut)

This is where Stucci gets its name. This is a typical wall structure with holes or bumps.

NoiseSize : 0.250

Turbulence: 5.00

NoiseSize (NumBut)

The dimensions of the Noise table.

Turbulence (NumBut)

The depth of the Stucci.

Soft noise

Hard noise

Soft Noise, Hard Noise (RowBut)

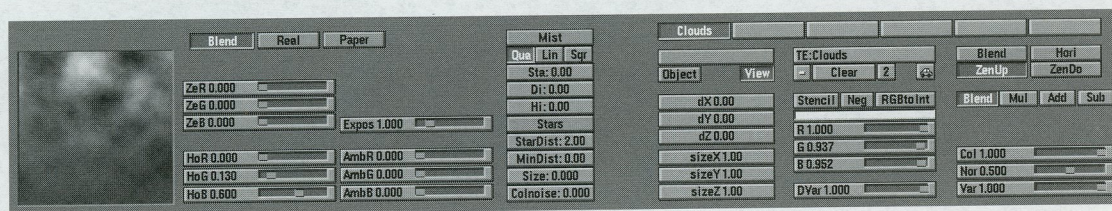
There are two methods available for working with Noise.

NOISE TEXTURE



Although this looks great, it is not *Perlin Noise*! This is true, randomly generated Noise. This gives a different result every time, for every frame, for every pixel.

The WorldButtons



The settings in this ButtonsWindow visualise the World DataBlock. It is linked to a Scene, and can therefore be reused by other Scenes.

This block contains the settings for standard backgrounds ('sky'), mist effects and the built-in star generator. The *ambient* colour and *exposure* time can be set here as well.

HotKey: F6.



The DataButtons in the Header indicate which World block is active.

World Browse (MenuBut)

Select another World from the list provided, or create a new block.

WO: (TextBut)

Give the current World block a new and unique name.

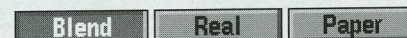
Users (But)

If the World block has more than one user this button shows the total number of users. Press the button to make the World "Single User". An exact copy is then created.

Remove Link (But)

Delete the link to the World.

Sky options.



Where the *alpha* in the rendering is less than 1.0, a *sky* colour is filled in. The *alpha* is then no longer usable for post-processing (unless the sky is black).

Blend (TogBut)

This option renders the background, e.g. a *sky*, with a natural progression. At the bottom of the image is the horizon colour. At the top, the colour of the zenith. The progression is not linear but bent in the shape of a ball, depending on the *lens* value of the Camera.

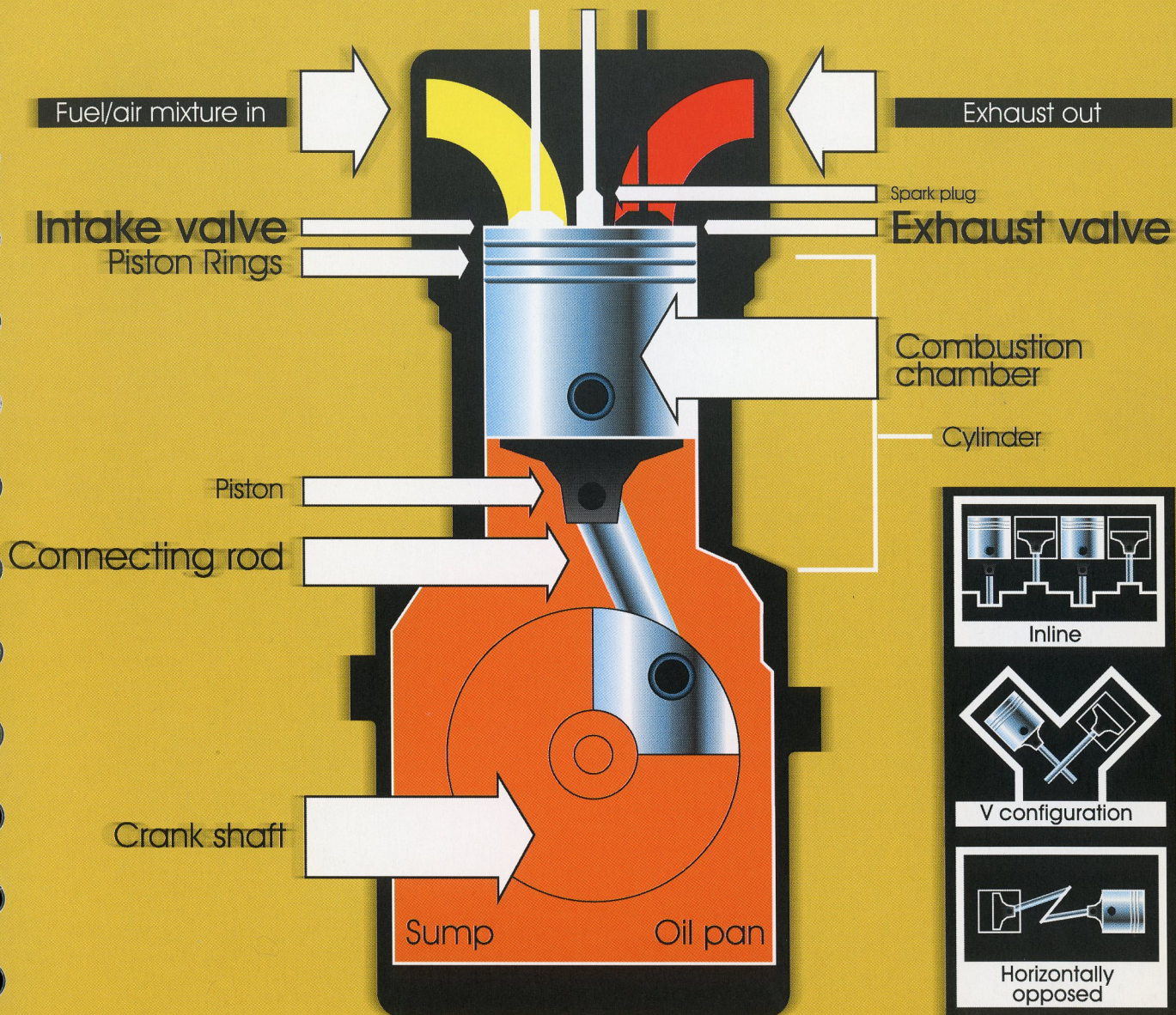
Real (TogBut)

The option "Real" makes the position of the horizon *real*; the direction in which the camera is pointed determines whether the horizon or the zenith can be seen. This also influences the generated texture coordinates.

Paper (TogBut)

This option makes the "Blend" (if this is selected) or the texture coordinates completely flat, at 'viewport' level.

Parts of an Engine





ZeR 0.000

ZeG 0.000

ZeB 0.000

HoR 0.000

HoG 0.130

HoB 0.600

ZeR, ZeG, ZeB (NumSli)

The colour of the *zenith*. This is the point directly above or directly below an observer (on the earth!).

HoR, HoG, HoB (NumSli)

The colour of the *horizon*.

AmbR 0.000

AmbG 0.000

AmbB 0.000

AmbR, AmbG, AmbB (NumSli)

The colour of the environmental light, the *ambient*. This is a rather primitive way to make the entire rendering lighter, or to change the colour temperature.

Expos 1.000

Expos (NumSli)

The lighting time, *exposure*. In fact, this causes a global strengthening or reduction in all the lamps. Use this to give the rendering more contrast.

Mist

Qua Lin Sqr

Sta: 0.00

Di: 0.00

Hi: 0.00

Mist (TogBut)

Activates the rendering of *mist*. All rendered faces and halos are given an extra *alpha* value, based on their distance from the camera. If a 'sky' colour is specified, this is filled in behind the *alpha*.

Qua, Lin, Sqr (RowBut)

Determines the progression of the mist. Quadratic, linear or inverse quadratic (square root), respectively. "Sqr" gives a thick 'soupy' mist, as if the picture is rendered under water.

Sta (NumBut)

The start distance of the mist, measured from the Camera.

Di (NumBut)

The depth of the mist, with the distance measured from "Sta".

Hi (NumBut)

With this option, the mist becomes thinner the higher it goes. This is measured from $Z = 0.0$. If the value of "Hi" is set to zero, this effect is turned off.

Stars

StarDist: 2.00

MinDist: 0.00

Size: 0.000

Colnoise: 0.000

Stars (TogBut)

Blender has an automatic star generator. These are standard halos that are only generated in the *sky*. With this option on, stars are also drawn in the 3D Window (as small points).

StarDist (NumBut)

The average distance between two stars. Do not allow this value to become too small, as this will generate an overflow.

MinDist (NumBut)

In actuality, stars are light years apart. In the Blender universe, this distance is much smaller. To prevent stars from appearing too close to the Camera, you can set a "MinDist" value. Stars will never appear within this distance.

Size (NumBut)

The average screen dimensions of a star.

ColNoise (NumBut)

This value randomly selects star colour

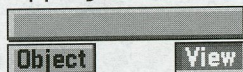
Texture Channels

Clouds

Texture name (RowBut)

A World has six *channels* with which Textures can be linked. This is only used for the *sky*. Each *channel* has its own *mapping*; i.e. the effect the texture has on the *sky*. The settings are in the buttons described below

Mapping: coordinates input.



Each Texture has a 3D coordinate (the texture coordinate) as input.

A sky has three options for this.

Object Name (TextBut)

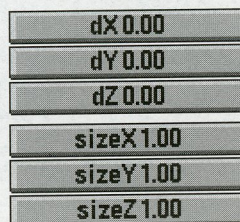
The name of the Object that is used as a source for the texture coordinates. If the Object does not exist, the button remains empty.

Object (RowBut)

Each Object in Blender can be used as a source for texture coordinates. To accomplish this, an inverse transformation is used to obtain the *local* Object coordinate. This links the texture to the position, dimensions and rotation of the Object.

View (RowBut)

The *view* vector of the camera is passed on to the texture.



Mapping: transform coordinates.

Use these buttons to more finely adjust the buttons texture coordinate.

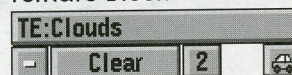
dX, dY, dZ (NumBut)

The extra translation of the texture coordinate.

sizeX, sizeY, sizeZ (NumBut)

The extra scaling of the texture coordinate.

Texture Block



TE: (TextBut)

The name of the Texture block. Use this button to change the name.

Texture Browse (MenuBut)

Choose an existing Texture from the list provided, or create a new Texture Block.

Clear (But)

The link to the Texture is erased.

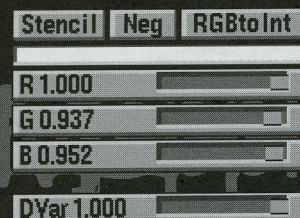
Users (But)

If the Texture Block has more than one user this button shows the total number of users. Press the button to make the Texture "Single User" An exact copy is then created.

Auto Name (But)

Blender assigns a name to the Texture.

Mapping: Texture input settings.



These buttons pass extra information on to the Texture.

Stencil (TogBut)

Textures are normally executed one after the other and layed over one another. A second Texture *channel* can completely replace the first one.

This option sets the *mapping to stencil* mode. No subsequent Texture can operate where this Texture is operating.

Neg (TogBut)

The Texture operation is reversed.

RGBtoInt (TogBut)

This option causes an RGB texture (works on colour) to be used as an Intensity texture (works on a value).

R, G, B (NumSli)

The colour that an Intensity texture blends with the current colour.

DVar (NumSli)

The value that an Intensity texture blends with the current value.

Mapping: output to.



Blend (TogBut)

The texture works on the colour progression in the sky.

Hori (TogBut)

The texture works on the colour of the horizon.

ZenUp (TogBut)

The texture works on the colour of the zenith above.

ZenDown (TogBut)

The texture works on the colour of the zenith below.

Mapping: output settings.



These buttons adjust the output of the Texture.

Blend (RowBut)

The Texture blends the values.

Mul (Rowbut)

The Texture multiplies the values.

Add (RowBut)

The Texture adds the values.

Sub (RowBut)

The Texture subtracts the values.

Col (NumSli)

The extent to which the texture works on colour.

Nor (NumSli)

The extent to which the texture works on the normal (not applicable here).

Var (NumSli)

The extent to which the texture works on a value (a single variable).

The AnimButtons

This ButtonsWindow visualises settings associated with animations, most of which are part of the Object DataBlock.

It can also be used to create Effects: the 'Build' and the 'Particles' Hotkey: F7.

OB:Plane

2

The typical 'browse' MenuBut is missing here. Link Objects to other Scenes with the LinkMenu (CTRL+L)

OB: (TextBut)

Give the Object block a new and unique name. The Object is inserted again, sorted alphabetically

Users (But)

If the Object block has multiple users, this button shows the total number of users. Press the button to make the Object "Single User" An exact copy is then created (exclusive the Object block).

Tracking buttons.

TrackX **Y** **Z** **-X** **-Y** **-Z** **UpX** **Y** **Z**

In Blender Objects can be assigned a rotation *constraint*:

Objects that always point in the direction of other Objects: CTRL+T or "Make Track"

Objects as Children of a Curve path, where the curve determines the rotation ("Follow" button)

- Particles can give rotations to Objects (see AnimButtons, Effects).

Because Objects have a rotation of their own, it is advisable to first erase this using ALT+R. If the Object is a Child, then erase the "Parent Inverse" as well using ALT+P.

Use these buttons to indicate how *tracking* must work:

TrackX, Y, Z, -X, -Y, -Z (RowBut)

Specifies the direction axis; the axis that, for example, must point to the other Object.

UpX, UpY, UpZ (RowBut)

Specify what axis must point 'up', in the direction of the (global) positive Z axis. If the "Track" axis is the same as the "Up" axis, this is turned off

Powertrack

PowerTrack (TogBut)

This option completely switches off the Object's own rotation and that of its Parents. Only for Objects that 'track' to another Object.

Draw Key

DrawKey (TogBut)

Draw Key Sel

If Objects have an Object lpo, they can be drawn in the 3Dwindow as *key positions*. Key positions are drawn with this option on and the lpoKeys on (in the lpoHeader). Hotkey: KKEY.

DrawKeySel (TogBut)

Limits the drawing of *Object keys* to those selected.

DupliFrames

DupSta: 1

DupEnd 100

No Speed

DupliVerts

DupOn: 1

DupOff 0

Duplicators

Blender can automatically generate Objects without actually creating them. To do this, an animation system must be created first. A 'virtual' copy of the Object will then be placed on every frame specified. It is also possible to have a virtual copy placed on each *vertex* (or particle).

This can be used as a modeling tool as well. To do this, select the duplicated Objects and press CTRL-SHIFT+A ("Make Dupli's Real").

DupliFrames (TogBut)

No matter how the Object moves, with its own Object lpos or on a Curve path, a copy of the Object is made for every frame from "DupSta" to "DupEnd". The "DupliFrames" system is built for the specified frame interval.

DupliVerts (TogBut)

Child Objects are duplicated on all vertices of this Object (only with Mesh).

DupSta, DupEnd (NumBut)

The start and end frame of the duplication.

DupOn, DupOff (NumBut)

Empty positions can be specified with the option "DupliFrames". For example: "DupOn" on '2', "DupOff" on '8' sets two copies on every 10 frames. The duplicated Objects move over the animation system like a sort of train.

No Speed (TogBut)

The "DupliFrames" are set to 'still', regardless of the current frame.

Slurph: 0

Slurph (NumBut)

This option is only available if there are VertexKeys.

The "Slurph" value specifies a fixed delay for the interpolation of Keys *per vertex*. The first vertex comes first, the last vertex has a delay of "Slurph" frames. This effect makes quite special and realistic Key framing possible. Watch the sequence of vertices carefully with Meshes. The sequence can be sorted with the commands EditButtons→Xsort and EditButtons→Hash. Naturally, it is important that this occurs *before* the VertexKeys are created, because otherwise quite unpredictable things can occur (however, it can be nice for Halos).

OffsOb

OffsPar

OffsParti

SlowPar

0.0000

TimeOffset: 0.00

Automatic Time

PrSpeed

OffsOb (TogBut)

The "TimeOffset" value works on its own Object lpo.

OffsPar (TogBut)

The "TimeOffset" value works on the Parent relationship of the Object.

OffsPart (TogBut)

The "TimeOffset" value works on the Particle Effect.

SlowPar (TogBut)

The value of "TimeOffset" is used to create a 'delay' in the Parent relationship. This delay is cumulative and depends on the previous frame. When rendering animations, the

complete sequence must always be rendered, starting with the first frame.

TimeOffset (NumBut)

Depending on the previously mentioned presets, the animation is shifted a number of frames. This does not work for VertexKeys.

Automatic Time (But)

This generates automatic "TimeOffset" values for all *selected* Objects. The start value is the value of the "TimeOffset" button. A requestor pops up and asks for the size of the interval. Blender looks at the Object's screen coordinates in the nearest 3DWindow and calculates the offset values from left to right.

PrSpeed (But)

The speed of the Object is printed.

Map Old: 100	Map New: 100
AnimSpeed: 4	
Sta: 1	End: 250

Map Old, Map New (NumBut)

This button can be used to modify the internal time calculation.

"Map Old" gives the previous value in frames; "Map New" specifies the number of frames that must be rendered. Only the mutual relations between these values are important.

Use this only to speed up or slow down the entire animation system. The absolute value 'frame' now becomes relative, which can be quite confusing if the animation must still be modified.

AnimSpeed (NumBut)

The maximum speed of the real-time animation playback, expressed in hundredths of a second.

Sta, End (NumBut)

The start and end frame of an animation to be rendered or played real-time.

PathLen: 100	CurvePath	CurveFollow	PrintLen
--------------	-----------	-------------	----------

These buttons are only displayed if the active Object is a Curve.

PathLen (NumBut)

The length of the Curve path in frames, if there is no Speed lpo.

CurvePath (TogBut)

Specifies that the Curve becomes a *path*. Children of this Curve now move over the curve. All Curves can become a *path*, but a 5th order Nurbs curve works best. It has no problems with movement and rotation discontinuity.

CurveFollow (TogBut)

The Curve path passes a rotation to the Child Objects. The 'Tracking' buttons determine which axis the path follows.

In EditMode, horizontal lines are also drawn for a 3D curve. This determines the *tilt*, which is an extra axis rotation of the Child Objects. The *tilt* can be changed using the **TKEY**.

Curve paths cannot give uniform perpendicular (aligned with the local Z axis) rotations. In that case, the 'up' axis cannot be determined.

PrintLen (But)

The length of the path is printed in Blender units.

63.000

64.000

Xmin: 63.00

Xmax: 64.00

-10.424

0.100

SET

Ymin: -10.42

Ymax: 0.10

These buttons are displayed if an IpoWindow is present in the same Screen.

Xmin, Xmax, Ymin, Ymax (NumBut)

The numbers above these buttons specify the boundingbox of all the visible curves in the IpoWindow. Use the buttons to enter a new value.

Set (But)

The new values of the boundingbox are assigned to the visible curves in the IpoWindow.

Speed: 0.10

SET

Speed (NumBut)

In certain cases, the exact speed of a translation caused by Object Ipos must be determined. Proceed as follows to do this:

- In the IpoWindow, make only the LocX, LoXy, LocZ curves visible.
- Set the IpoKey option ON (KKEY in the IpoWindow).

Select the keys that must be assigned a particular speed.

- Only keys that already have a speed and direction can be changed. If the speed is 0.0, nothing happens.
- Press the "Set" Button.

ANIM EFFECTS: BUILD

NEW Effect

Delete

Build

Effects will be supplemented in a subsequent Blender release by a Plugin System. Two are currently built in: "Build" and "Particles". Effects are a fixed part of the Object; they cannot have any links or multiple users.

New Effect (But)

Create a new Effect.

Delete (But)

Delete the Effect.

Build (MenuBut)

Select an effect.

Len: 100.00

SFra: 1.00

The Build Effect works on Meshes, which are built up face by face over time. It also works on the vertices in Halo Meshes.

The sequence in which this happens can be specified in the 3DWindow with CTRL+F: "Sort Faces" (not in EditMode). The *faces* of the *active* Mesh Object are sorted. The current face in the 3DWindow is taken as the starting point. The leftmost *face* first, the rightmost *face* last.

Len (NumBut)

The total time the building requires.

SFra (NumBut)

The frame number on which the Effect starts.

ANIM EFFECTS: PARTICLES

Particles are halos (or Objects if the option "DupliVerts" is ON) that are generated more or less according to laws of physics. Use Particles for smoke, fire, explosions, a fountain, fireworks or a school of fish!

A Particle system is pre-calculated as a pre-process (this can take some time). They can then be viewed in the 3DWindow in real time.

Particles are a full-fledged part of Blender's animation system. They can be also be controlled and by Lattices. Only Meshes can have Particles.

Tot: 1000 Sta: 1.00 End: 100.00 Life: 50.00 Keys: 8

Tot (NumBut)

The total number of Particles. Particles require quite a bit of memory (not in the file!) and rendering time, so specify this value carefully.

Sta, End (NumBut)

The start and end frame between which Particles are generated.

Life (NumBut)

The life span of each Particle.

Keys (NumBut)

Not all Particle locations are calculated and remembered for each frame for the entire particle system. This is only done for a fixed number of *key* positions between which interpolations are performed. A larger number of "Keys" gives a more fluid, detailed movement. This makes significant

demands on the memory and time required to calculate the system.

CurMul: 0 Mat: 1 Mult: 0.000 Life: 50.00 Child: 4

CurMul (NumBut)

Particles can 'multiply themselves' at the end of their lives. For each generation, certain particle settings are unique. This button determines which generation is displayed.

Mat (NumBut)

The Material used for the current generation of Particles.

Mult (NumBut)

This determiness whether the particles multiply themselves. A value of 0.0 switches this off. A value of 1.0 means that each Particle multiplies itself

The particle system itself ensures that the *total* number of Particles is limited to the "Tot" value.

Life (NumBut)

The age of the Particles in the following generation.

Child (NumBut)

The number of children of a Particle that has multiplied itself

RandLife: 0.000 Seed: 0 Bspline Vect VectSize 0.000

RandLife (NumBut)

A factor that ascribes the age of Particles a (pseudo) random variation.

Seed (NumBut)

The offset in the random table.

Bspline (TogBut)

The Particles are interpolated from the *keys* using a B-spline formula. This give a much

more fluid progression, but the particles no longer pass exactly through the *key* positions.

Vect (TogBut)

This gives particles a rotation direction. This can be seen in the Halo rendering. Particles that duplicate Objects now also give a rotation to these Objects.

VectSize (TogBut)

The extent to which the speed of the "Vect" Particle works on the dimensions of the Halo.

Norm: 0.000 Ob: 0.000 Rand: 0.100 Tex: 0.000 Damp: 0.000

Norm (NumBut)

The extent to which the vertex normal of the Mesh gives the Particle a starting speed. If the Mesh has no faces (and thus no vertex normals) the normalised *local* vertex coordinate is used as the starting speed.

Ob (NumBut)

The Extent to which the speed of the Object gives the Particle a starting speed. This makes a rotating cube become a sort of 'sprinkler'.

Rand (NumBut)

The extent to which a (pseudo) random value gives the Particle a starting speed.

Tex (NumBut)

The extent to which the Texture gives the Particle a starting speed. For this, only the last Texture of the Material is used, in *channel* number 8.

Damp (NumBut)

Use of damping reduces the speed, like a sort of friction.

X: 0.000

Y: 0.000

Force:

Z: 0.000

Force X, Y, Z (NumBut)

A standard, continually present force. This can simulate the effect of gravity or wind.

X: 0.000

Y: 0.000

Texture:

Z: 1.000

Texture X, Y, Z (NumBut)

A standard force that works on a Particle, determined by the texture.

Int

RGB

Grad

Nabla: 0.050

Textures can have an effect on the movement of Particles. The 3D coordinate of the Particle is passed to the texture per Particle *key*.

Int (RowBut)

The Intensity that is passed back from the texture is used as a factor for the standard texture force (previous three buttons).

RGB (RowBut)

The colour of the texture has a direct effect on the speed of the Particle: Red on the X, Green on the Y and Blue on the Z component of the speed.

Grad (RowBut)

The *gradient* of the texture is calculated. This is the mathematical derivative. Four samples of the texture are combined to produce a speed vector. With procedural textures, such as Clouds, this method gives a very beautiful, turbulent effect. Set the number of "Keys" as high as possible to see the sometimes rather subtle twisting.

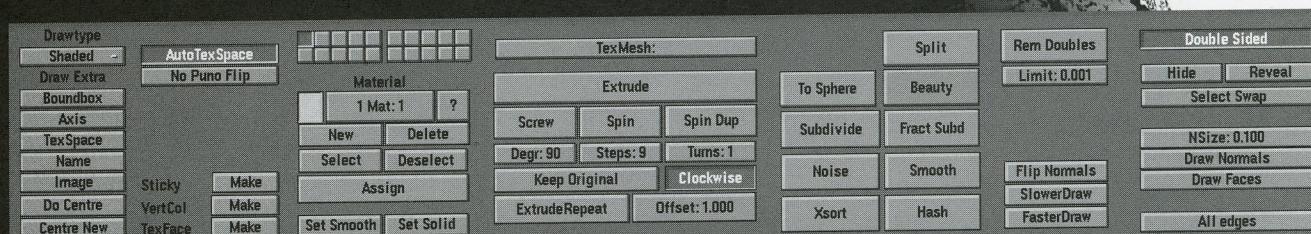
Anti Buttons

Nabla (NumBut)

The dimension of the area in which the *gradient* is calculated. This value must be carefully adjusted to the frequency of the texture.

The EditButtons

EDITBUTTONS, GENERAL



The settings in this ButtonsWindow visualise the ObData blocks and provide tools for the specific EditModes. Certain buttons are redrawn depending on the type of ObData. The types are: Mesh, Curve, Surface, Text, MetaBall, Lattice, Ika and Camera. This section describes the buttons that appear for nearly all ObData. Later in the text, the buttons are grouped per ObData type. A complete overview of all HotKeys for EditMode is provided in the 3Dwindow section. HotKey: F9.

ME:Plane 2

The DataButtons in the Header specify what block is visualised. Mesh is used as an example here, but the usage of the other types of ObData is identical.

Mesh Browse (MenuBut)

Select another Mesh from the list provided.

ME: (TextBut)

Give the current block a new and unique name. The new name is inserted in the list, sorted alphabetically.

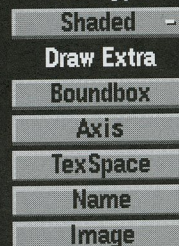
Users (But)

If the block is used by more than one Object, this button shows the total number of Objects.

Press the button to change this to "Single User". An exact copy is then created.

This group of buttons specifies Object characteristics. They are displayed here for ease.

Drawtype



DrawType (MenuBut)

Choose a preference for the standard display method in the 3D window from the list provided. The "DrawType" is compared with the "DrawMode" set in the 3Dheader; the least complex method is the one actually used.

The types, in increasing

degree of complexity, are:

- BoundBox. A cube in the dimensions of the object is drawn.
- Wire. The wire model is drawn.
- Solid. Zbuffered with the standard OpenGL lighting
- Shaded. This display, which uses Gouraud shading, is the best possible approach to the manner in which Blender renders. It depicts the situation of a single frame from the Camera. Use CTRL+Z to force a recalculation.

The "Draw Extra" options are displayed atop the selected DrawType.

BoundingBox (TogBut)

A cube is displayed in the dimensions of the object.

Axis (TogBut)

The axes are drawn with X, Y and Z indicated.

TexSpace (TogBut)

The texture space. This can be different from the BoundingBox. It is displayed with broken lines.

Name (TogBut)

The name of the Object is printed at the Object centre.

Image (TogBut)

Draws the (texture) Image as wire. This option has a limited function. It can only be used for 2D composing
→156

Do Centre

Centre New

Do Centre (But)

Each ObData has its own local 3D space. The null point of this space is placed at the Object centre. This option calculates a new centred null point in the ObData. This may change texture coordinates.

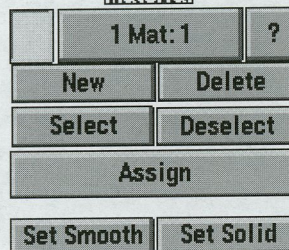
Centre New (But)

As above, but now the Object is placed in such a way that the ObData appears to remain in the same place.



The layer setting of the Object. Use SHIFT+LeftMouse to activate multiple layers.

Material



Material indices.

Objects and ObData can be linked to more than one Material. This can be managed with these buttons.

1 Mat 1 (NumBut)

This button can be used to specify which Material should be shown, i.e. which Material is active. The first digit indicates the amount of Materials, the second digit indicates the index number of the active Material. Each face in a Mesh has a corresponding number: the 'Material index'. The same is true of Curves and Surfaces.

? (But)

In EditMode, this Button indicates what index number and thus what Material, the selected items have.

New (But)

Make a new index. The current Material is assigned an extra link. If there was no Material, a new one is created.

Delete (But)

Delete the current index. The current Material gets one less link. The already used index numbers are modified in the ObData.

Select (But)

In EditMode, everything is selected with the current *index* number.

Deselect (But)

In EditMode, everything is deselected with the current *index* number.

Assign (But)

In EditMode, the current *index* number is assigned to the selected items.

Set Smooth (But)

This sets a *flag* which specifies that rendering must be performed with normal interpolation. In EditMode, it works on the selection. Outside EditMode everything becomes "Smooth".

Set Solid (But)

This sets a *flag* which indicates that rendering must be "Solid". In EditMode this works on the selection. Outside EditMode everything becomes "Solid".

EDITBUTTONS, MESH

AutoTexSpace

No Puno Flip

AutoTexSpace (TogBut)

This option calculates the texture area automatically, after leaving EditMode. You can also specify a texture area yourself (Outside EditMode, in the 3DWindow; **TKEY**), in which case this option is turned OFF.

No Puno Flip (TogBut)

In Blender's turbo language, 'Punos' are the vertex normals. Because Blender normally renders double-sided, the direction of the normal (towards the front or the back) is automatically corrected during rendering. This option turns this automatic correction off,

allowing "smooth" rendering with faces that have sharp angles (smaller than 100 degrees). Be sure the face normals are set consistently in the same direction (**CTRL+N** in EditMode).

Sticky

Make

VertCol

Make

Make Sticky (But)

Blender allows you to assign a texture coordinate to Meshes that is derived from the way the Camera view sees the Mesh. The screen coordinates (only X,Y) are calculated from each vertex and these coordinates are stored in the Mesh. As if the texture is permanently projected and fixed on the Mesh as viewed from the Camera; it becomes "sticky". Use "Sticky" to match a 3D object exactly with the Image Texture of a 3D picture. This option also allows you to create special *morphing* effects. If the image is already "sticky", the button allows you to remove this effect.

Make VertCol (but)

A colour can be specified per vertex. This is required for the VertexPaint option. If the Object DrawType is "Shaded", these colours are copied to the vertex colours. This allows you to achieve a *radiosity-like* effect (set MaterialButtons→VertCol on). If the Mesh is "Double Sided", this is automatically turned off.

→155

TexMesh:

TexMesh (TextBut)

Enter the name of another Mesh block here to be used as the source for the texture coordinates. *Morphing-like* effects can then be achieved by distorting the *active* Mesh. For example, a straight stream of water (as an animated texture) can be placed in a winding river.

Editing Buttons

Extrude		
Screw	Spin	Spin Dup
Degr: 90	Steps: 9	Turns: 1
Keep Original		Clockwise
ExtrudeRepeat		Offset: 1.000

Extrude (But)

The most important of the Mesh tools: Extrude Selected. "Extrude" in EditMode converts all selected *edges* to *faces*. If possible, the selected faces are also duplicated.

Grab mode starts immediately after this command is executed.

If there are multiple 3DWindows, the mouse cursor changes to a question mark. Click at the 3DWindow in which "Extrude" must be executed.

HotKey: EKEY.

Screw (But)

This tool starts a repetitive "Spin" with a screw-shaped revolution. You can use this to create screws, springs or shell-shaped structures. The method for using the "Screw" function is strict:

Set the 3DWindow in *front view*, PAD_1.

Place the 3DCursor at the position through which the rotation axis must pass, vertically on the screen.

Ensure that an *open poly line* is always available. This can be a single edge or half a circle, or ensure that there are two 'free' ends; two edges with only one vertex linked to another edge. The

"Screw" function localises these two points and uses them to calculate the vector that is added to the "Spin" per full rotation. If these two vertices are at the same location, this creates a normal "Spin". Otherwise, interesting things happen!

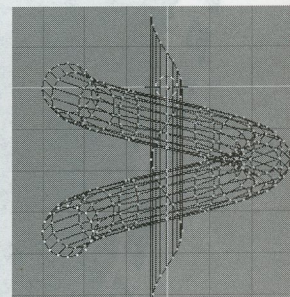
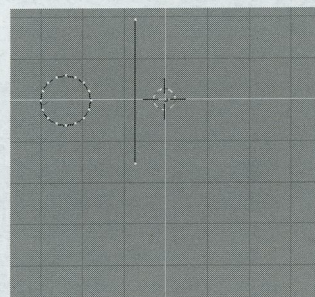
Select all vertices that must participate in the "Screw"

Give the buttons "Steps" and "Turns" the desired value.

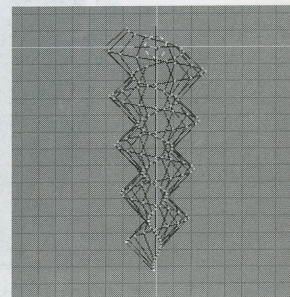
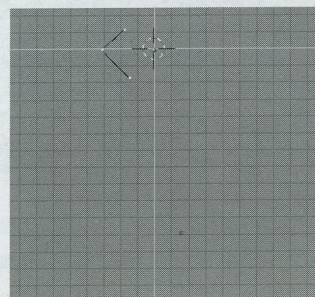
Press "Screw"!

If there are multiple 3DWindows, the mouse cursor changes to a question mark. Click at the 3DWindow in which "Extrude" must be executed.

HotKey: WKEY.



These two images show how to make a spring



and these how to make a screw.

28 m

30 m

LITERS
6 7 8 9





Spin (But)

The "Spin" operation is a repetitively rotating "Extrude". This can be used in every view of the 3DWindow, the rotation axis is always through the 3DCursor, perpendicular to the screen. Set the buttons "Degr" and "Steps" to the desired value.

If there are multiple 3DWindows, the mouse cursor changes to a question mark. Click at the 3DWindow in which the "Spin" must occur.

Spin Dup (But)

Like "Spin", but instead of an "Extrude", there is duplication.

HotKey: WKEY.

Degr (NumBut)

The number of degrees the "Spin" revolves.

Steps (NumBut)

The total number of "Spin" revolutions, or the number of steps of the "Screw" per revolution.

Turns (NumBut)

The number of revolutions the "Screw" turns.

Keep Original (TogBut)

This option saves the selected original for a "Spin" or "Screw" operation. This releases the new vertices and faces from the original piece.

Clockwise (TogBut)

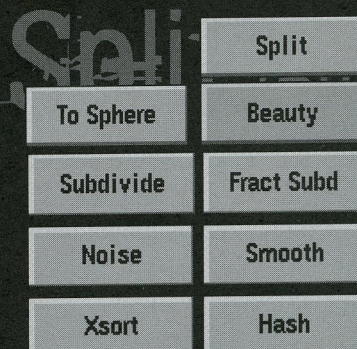
The direction of the "Screw" or "Spin", clockwise, or counterclockwise.

Extrude Repeat (But)

This creates a repetitive "Extrude" along a straight line. This takes place perpendicular to the view of the 3DWindow.

Offset (NumBut)

The distance between each step of the "Extrude Repeat". HotKey: WKEY.

**Split (But)**

In EditMode, this command 'splits' the selected part of a Mesh without removing faces. The split sections are no longer connected by edges. Use this to control *smoothing*.

Since the split parts can have vertices at the same position, we recommend that you make selections with the LKEY.

HotKey: YKEY.

To Sphere (But)

All selected vertices are blown up into a spherical shape, with the 3DCursor as a midpoint. A requester asks you to specify the factor for this action.

HotKey: WKEY.

Beauty (TogBut)

This is an option for "Subdivide". It splits the faces into halves lengthwise, converting elongated faces to squares. If the face is smaller than the value of "Limit", it is not longer split in two.

Subdivide (But)

Selected faces are divided into quarters; all edges are split in half.

HotKey: WKEY.

Buttons

Fract Subd (But)

Fractal Subdivide. Like "Subdivide" but now the new vertices are set with a random vector up or down. A requestor asks you to specify the amount. Use this to generate landscapes or mountains.

Noise (But)

Here Textures can be used to move the selected vertices up a specific amount. The local vertex coordinate is used as the texture coordinate. Every Texture type works with this option. For example, the Stucci produce a landscape effect. Or use Images to express this in relief

Smooth (But)

All edges with both vertices selected are shortened. This flattens sharp angles.
HotKey: wKEY.

Xsort (But)

Sorts the vertices in the X direction. This creates interesting effects with VertexKeys or 'Build Effects' for Halos.

Hash (But)

This makes the sequence of vertices completely random.

Rem Doubles

Limit: 0.001

Rem Doubles (But)

Remove Doubles. All selected vertices closer to one another than "Limit" are combined and redundant faces are removed.

Flip Normals

SlowerDraw

FasterDraw

Flip Normals (But)

Toggles the direction of the face normals.
HotKey: wKEY.

SlowerDraw, FasterDraw (But)

When leaving EditMode all edges are tested to determine whether they must be displayed as a wire frame. Edges that share two faces with the same normal are never displayed. This increases the recognisability of the Mesh and considerably speeds up drawing. With "SlowerDraw" and "FasterDraw" you can specify that additional or fewer edges must be drawn when you are not in EditMode.

Double Sided

Hide

Reveal

Select Swap

NSize: 0.100

Draw Normals

Draw Faces

All edges

Double Sided (TogBut)

Only for display in the 3Dwindow; can be used to control whether double-sided faces are drawn. Turn this option off if the Object has a negative 'size' value (for example an X-flip)

Hide (But)

All selected vertices are temporarily hidden.
HotKey: hKEY.

Reveal (But)

This undoes the "Hide" option. HotKey: ALT+H.

Select Swap (But)

Toggle the selection status of all vertices.

NSize (NumBut)

The length of the face normals, if they have been drawn.

Draw Normals (NumBut)

Indicates that the face normals must be drawn in EditMode.

Draw Faces (NumBut)

Indicates that the face must be drawn (as Wire) in EditMode. Now it also indicates whether faces are selected.

AllEdges (NumBut)

After leaving EditMode, all edges are drawn normally, without optimisation.

Nurb (But)

A Nurbs curve is mathematically quite 'pure'. For example: it can be used to create perfect circles.

Make Knots

Uniform U	Y
Endpoint U	Y
Bezier U	Y

Nurbs curves have *knots*, a row of numbers that specify the exact curve. Blender offers three pre-sets for this:

Uniform U, V (But)

Sets the *knots* to create a uniform distribution. Use this for closed curves or surfaces.

Endpoint U, V (But)

Sets the *knots* so that the first and last vertices are always included.

Bezier U, V (But)

Sets the *knots* table in such a way that the Nurbs behave like a Bezier.

Order U: 5	Y: 0
Resol U: 32	V: 12

Order U, V (NumBut)

The *order* is the 'depth' of the curve calculation. Order '1' is a point, order '2' is linear, order '3' is quadratic, etc. Always use *order* '5' for Curve paths. Order '5' behaves fluently under all circumstances, without annoying discontinuity in the movement.

ResolU, V (NumBut)

The resolution in which the interpolation occurs; the number of points that must be generated between two vertices in the curve.

EDITBUTTONS, CURVE AND SURFACE

Convert

These options convert selected curves.

Poly
Bezier
Bspline
Cardinal
Nurb

Poly (But)

A polygon only gives a linear interpolation of vertices.

Bezier (But)

Vertices in a Bezier curve are grouped in threes; the *handles*. The most frequently used curve type for creating letters or logos.

Bspline (But)

Obsolete.

Cardinal (But)

Obsolete.

Set Weight

Weight: 1.000

1.0

sqrt(2)/4

sqrt(3)/9

Set Weight (But)

Nurbs curves have a 'weight' per vertex; the extent to which a vertex participates in the interpolation. This button assigns the "Weight" value to all selected vertices.

Weight (But)

The weight that is assigned with "Set Weight"

1.0, sqrt(2)/4, sqrt(3)/9 (But)

A number of pre-sets that can be used to create pure circles and spheres.

→103

DefResolU: 6

Set

Back

Front

3D

DefResolU (NumBut)

The standard resolution in the U direction for curves.

Set (But)

Assigns the value of "DefResolU" to all selected curves.

Back (TogBut)

Specifies that the back side of (extruded) 2D curves should be filled.

Front (TogBut)

Specifies that the front side of (extruded) 2D curves should be filled.

3D (TogBut)

The curve may now have vertices on each 3D

coordinate; the front and back side are never rendered.

Width: 1.000

Ext1: 0.000

Ext2: 0.000

BevResol: 0

BevOb:

These buttons are only drawn for Curve and Font Objects.

Width (NumBut)

The interpolated points of a curve can be moved farther apart or brought closer together

Ext1 (NumBut)

The depth of the extrude.

Ext2 (NumBut)

The depth of the standard bevel

BevResol (NumBut)

The resolution of the standard bevel; the bevel eventually becomes a quarter circle.

BevOb (TextBut)

The bevel Object. Fill in the name of another Curve Object; this now forms the bevel. For each interpolated point on the curve, the bevel Object' is, as it were, extruded and rotated. With this method, for example, you can create the rails of a roller coaster with a 3D curve as the base and two small squares as bevels. Set the values "ResolU" of both Curves carefully given that this beveling can generate many faces.

Hide (But)

All selected vertices are hidden temporarily

Reveal (But)

This undoes the "Hide" operation.

Select Swap (But)

Toggle the selection status of all vertices.

Subdivide (But)

Create new vertices or *handles* in curves.

NSize (NumBut)

This determines the length of the 'tilt' lines in 3DCurves.

Spin

Spin (But)

This button is only available for Surface Objects. It makes selected Nurb curves a surface of revolution. The rotation axis runs perpendicular to the screen through the 3DCursor.

EDITBUTTONS, FONT

Bfont

Load Font

ToUpper

Font type (MenuBut)

Select a font from the list.

Load Font (But)

In the FileSelect Window, this specifies an "Adobe type 1" file that must be read.

ToUpper (But)

In EditMode, changes all letters into capitals or, if there are no small letters, changes all capitals to small letters.

Left

Right

Middle

Flush

TextOnCurve:

Left (RowBut)

All text is left-aligned.

Right (RowBut)

All text is right-aligned.

Middle (RowBut)

The text is centered.

Flush (RowBut)

The text is spread out to full length; the length of the longest sentence.

TextOnCurve (TextBut)

Enter the name of a Curve Object here; this now forms the line along which the text is placed.

Size: 1.000

Linedist: 1.000

Spacing: 1.000

Y offset: 0.00

Shear: 0.000

X offset: 0.00

Size (NumBut)

The letter size.

Linedist (NumBut)

The distance between two lines of text.

Spacing (NumBut)

The size of the space between two letters.

Yoffset (NumBut)

This shifts the text up or down. For adjusting "TextOnCurve"

Shear (NumBut)

Changes the letters to italics.

Xoffset (NumBut)

This moves the text left or right. For adjusting "TextOnCurve"

Ob Family:

Ob Family (TextBut)

You can create fonts yourself within a Blender file. Each letter from this Font Object is then replaced by any Object you chose, and is

automatically duplicated. This means that you can type with Objects!

Objects to be considered as letters must belong to the same 'family'; they must have a name that corresponds to the other letter Objects and with the name that must be entered in this button.

Important: set the option

AnimButtons→DupliVerts ON!

For example:

"Ob Family" = Weird.

The Objects that are to replace the letters a and b are called 'Weirda' and 'Weirdb' respectively.

DefResolU: 6 Set

Back Front 3D

Width: 1.000
Ext1: 0.000
Ext2: 0.000
BevResol: 0
BevOb:

These buttons were discussed in the previous section.

EDITBUTTONS, METABALL

WireSize:0.400
RenderSize:0.200
Threshold:0.600

WireSize (NumSli)

Determines the resolution of the MetaBall displayed in the 3DWindow. Be careful with small values, as they use a lot of memory

RenderSize (NumSli)

The resolution of the rendered MetaBall.

Threshold (NumSli)

This value determines the global 'stiffness' of the MetaBall.

Update:

Always
Half Res
Fast

Always (RowBut)

In EditMode, the MetaBall is completely recalculated during transformations.

HalfRes (RowBut)

The MetaBall is calculated in half resolution.

Fast (RowBut)

The MetaBall is only recalculated after the transformation.

Stiffness:2.000
Len:1.00

Stiffness (NumSli)

The stiffness can be specified separately per ball only for the active ball.

Len (NumSli)

MetaBall elements can also be tube-shaped. This button specifies the length of the *active* ball.

Negative

Ball

TubeX

TubeY

TubeZ

Negative (Tog)

The active 'ball' has a negative effect on the other balls.

Ball (RowBut)

The standard type.

TubeX, TubeY, TubeZ (RowBut)

The active 'ball' becomes a tube; in the X, Y or Z direction.

Make Regular

Outside

Make Regular (But)

This option sets the Lattice in a regular, standard position.

Outside (TogBut)

This type Lattice only displays vertices on the outside. The inner vertices are interpolated linearly. This makes working with large Lattices simpler.

EDITBUTTONS, IKA

Set Reference

Mem 0.300

Iter: 5

Set Reference (But)

The reference position of an Ika determines the position from which the Ika is calculated towards the user-specified position. This results in a sort of 'memory', a rest mode to which the Ika can always return. A slightly bent form works best as a reference.

This position is also evaluated if the Ika has a Skeleton deformation; this is the state in which *no* deformation occurs.

Mem (NumSli)

This is the extent to which the reference position has an effect on the Ika setting. Set this value to 0.0 to create a completely slack chain.

Iter (Num)

The number of iterations of the Ika calculation. To achieve a natural expression, this value can be kept low. During animation or

EDITBUTTONS, LATTICE

Meshes and Surfaces can be deformed with Lattices, provided the Lattice is the Parent of the Mesh or Surface.

U: 5	Lin	Card	B
V: 4	Lin	Card	B
W: 4	Lin	Card	B

U, V, W (NumBut)

The three dimensions of the Lattice. If a new value is entered here, the Lattice is placed in a regular, standard position.

Lin, Card, B (NumBut)

The manner in which the deformation is calculated can be specified per dimension of the Lattice. The options are: Linear, Cardinal spline and B spline. The last option gives the most fluid deformation.

transformation, the Ika then moves to the desired position slowly.

Limb Weight

Limb 0: 0.010

Limb 1: 0.010

Limb 2: 0.010

Limb Weight (NumBut)

These numbers give a relationship factor per *limb* for how stiff or heavy the limb is in relation to other limbs.

Skeleton Weight

Ika (0): 1.000

Ika (1): 1.000

Ika (2): 1.000

Skeleton Weight (NumBut)

A Skeleton deformation can consist of multiple Ika Objects. These numbers determine the extent to which each Ika contributes to deformation.

EDITBUTTONS, CAMERA

Lens: 35.00

ClipSta: 0.10

ClipEnd: 100.00

Lens (NumBut)

This number is derived from the lens values of a photo camera: 120' is telelens, '50' is normal, 28' is wide angle.

ClipSta, ClipEnd (NumBut)

Everything that is visible from the Camera's point of view between these values is rendered. Try to keep

these values close to one another so that the Zbuffer functions optimally

DrawSize: 0.500

Ortho

ShowLimits

Show Mist

DrawSize (NumBut)

The size in which the Camera is drawn in the 3DWindow

Ortho (TogBut)

A Camera can also render orthogonally. The distance from the Camera then has no effect on the size of the rendered objects.

ShowLimits (TogBut)

A line that indicates the values of "ClipSta" and "ClipEnd" is drawn in the 3Dwindow near the Camera.

ShowMist (TogBut)

A line that indicates the area of the 'mist' (see WorldButtons) is drawn near the Camera in the 3Dwindow.

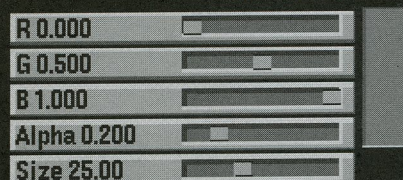
The VertexPaint Buttons

In Blender, the vertices of a Mesh can be assigned a colour, using EditButtons→"Make VertCol". Then, you can change the colour manually, as if you are painting the Mesh (start vertexPaint mode with vkey in the 3DWindow).

This ButtonsMenu has no HotKey, and can only be invoked in the ButtonsHeader with the 'brush' IconBut.

→155

Vertex Paint



R, G, B (NumSli)

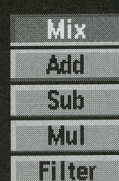
The active colour used for painting.

Alpha (NumSli)

The extent to which the vertex colour changes while you are painting.

Size (NumSli)

The size of the brush, which is drawn as a circle during painting.



Mix (RowBut)

The manner in which the new colour replaces the old when painting: the colours are mixed.

Add (RowBut)

The colours are added.

Sub (RowBut)

The paint colour is subtracted from the vertex colour.

Mul (RowBut)

The paint colour is multiplied by the vertex colour.

Filter (RowBut)

The colours of the vertices of the painted face are mixed together, with an "alpha" factor.

Area	Soft	Normals
Set	Mul: 1.00	Gamma: 1.000

Area (TogBut)

In the back buffer, Blender creates an image of the painted Mesh, assigning each face a colour number. This allows the software to quickly see what faces are being painted. Then, the software calculates how much of the face the brush covers, for the degree to which paint is being applied. You can set this calculation with the option "Area".

Soft (TogBut)

This specifies that the extent to which the vertices lie within the brush also determine the brush's effect.

Normals (TogBut)

The vertex normal (helps) determine the extent of painting. This causes an effect as if painting with light.

Set (But)

The "Mul" and "Gamma" factors are applied to the vertex colours of the Mesh.

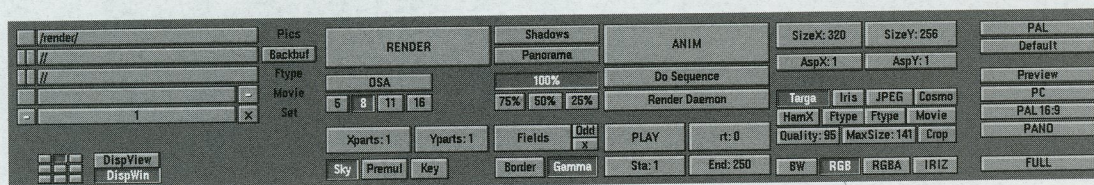
Mul (NumBut)

The number by which the vertex colours can be multiplied.

Gamma (NumBut)

The number by which the clarity of the vertex colours can be changed.

The DisplayButtons



This button field contains all the settings and commands that involve displaying and rendering. All of these data are part of the Scene block. They must thus be set separately for each Scene within a Blender file.

Hotkey: F10.

Pics (TextBut)

Enter the name of the directory to which the rendered image must be written if the "ANIM" command is given and, where required, the first few letters of the file name. Blender automatically assigns files a number frame 1 becoming 0001.

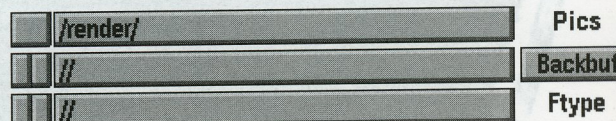
In this example, pictures are written as the files /render/0001, /render/0002 etc. If you enter "/render/rt" in the button, the files will be called /render/rt0001, /render/rt0002.

Blender creates the specified directories itself if they do not already exist.

The small square button to the left of the TextBut is used to invoke a FileSelect. Use it to select the output directory, and possibly a file name, and press ENTER to assign this to the TextBut.

Backbuf (TextBut)

Assigns a file name to the picture to be used as the background. Blender uses



the rendered *alpha* to determine the extent to which this background is visible.

A code can be processed into the file name, which allows an already rendered animation to be used as a background. Be sure that a '#' is placed at the end. This is replaced by the current (four-digit) frame number. For example: /render/work/rt.# becomes /render/work/rt.0101 at frame 101.

The two small buttons to the left of the TextBut invoke an ImageSelect or a FileSelect Window. Specify the file and press ENTER to assign it to the TextBut.

BackBuf (TogBut)

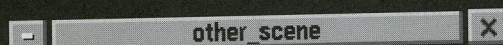
Activate the use of a background picture.

Ftype (TextBut)

Use an "Ftype" file, to indicate that this file serves as an example for the type of graphics format in which Blender must save images. This method allows you to process 'color map' formats. The colormap data are read from the file and used to convert the available 24 or 32

bit graphics. If the option "RGBA" is specified, standard colour number '0' is used as the transparent colour.

Blender reads and writes (Amiga) IFF, Targa, (SGI) Iris and CDi RLE colormap formats. Here, as well, the two small buttons to the left of the TextBut can be used to invoke an ImageSelect or a FileSelect window.



Each Scene can use another Scene as a "Set". This specifies that the two Scenes must be integrated when rendered. The current Lamps then have an effect on both Scenes. The render settings of the Set, such as the Camera, the *layers* and the World used, are replaced by those of the current Scene. Objects that are linked to both Scenes are only rendered once.

A Set is displayed with light gray lines in the 3DWindow. Objects from a Set cannot be selected here.



Window Location (TogBut)
These nine buttons visualise the standard position in which the Render Window appears on the screen.

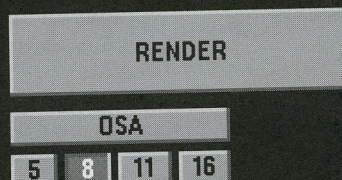
The setting in the example specifies that the Window must be at the top in the middle.

DispView (TogBut)

The rendering can also be displayed in the 3Dwindow instead of in a separate window. This occurs, independent of the resolution, precisely within the render borders of Camera view. Use F11 to remove or recall this image.

DispWin (TogBut)

The rendering occurs in a separate window. Use F11 to move this window to the foreground or background.



RENDER (But)

Start the rendering. This is a 'blocking' process. A square mouse cursor indicates that Blender is busy. Hotkey: F12.

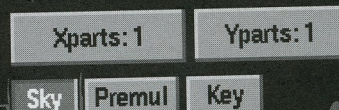
Rendering can also take place in the 'background'. Use the command line for this: `blender -b file.blend -f "framenummer"`.

OSA (TogBut)

OverSampling. This option turns anti-aliasing on, to achieve 'soft' edges and perfectly displayed Image textures. OSA rendering generally takes 1.5 to 2 times longer than normal rendering.

5, 8, 11, 16 (RowBut)

Blender uses a Delta Accumulation rendering system with *jittered sampling*. These numbers are pre-sets that specify the number of *samples*; a higher value produces better *edges*, but slows down the rendering.



Xparts, Yparts (NumBut)

OSA rendering of large images, in particular, can take up a lot of memory. In addition to all the shadow buffers and texture maps and the faces themselves, this takes up 10 to 16 bytes

per pixel. For a 2048*1024 picture, this requires a minimum of 32 Mb free memory

Use this option to subdivide the rendering into 'parts' Each part is rendered independently and then the parts are combined. The "Xparts" are particularly important when rendering "Ztransp" faces.

Sky (RowBut)

If a World has 'sky' this is filled in in the background. The alpha is not altered, but the transparent colours 'contaminate' the background colours, which makes the image less suitable for post-processing

Premul (RowBut)

Sky' is never filled in. The *alpha* in the picture is delivered as "Premul": a white pixel with *alpha* value 0.5 becomes: (RGBA bytes) 128, 128, 128, 128. The colour values are thus multiplied by the *alpha* value in advance.

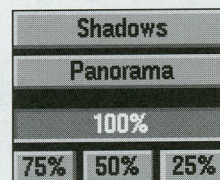
Use "Premul" *alpha* for post-processing such as filtering or *scaling*. Remember to select the "RGBA" option before saving.

When Blender reads RGBA files, "Premul" is considered the standard.

Key (RowBut)

Sky' is never filled in. The *alpha* and colour values remain unchanged. A white pixel with an *alpha* value of 0.5 becomes: (RGBA bytes) 255, 255, 255, 128. What this means is especially clear when rendering Halos: the complete transparency information is in the (hidden) *alpha* layer

Many drawing programs work better with "Key" *alpha*.



Shadows (TogBut)

This turns shadow rendering on. Shadow can only be created by Spot Lamps.

Panorama (TogBut)

Blender can render panoramas. To do this, a number of pictures are rendered, where the number in question corresponds with the value of "Xparts" For each 'part' the Camera is rotated in such a way that a continuous panorama is created.

For the width of the panorama of the Camera Lens, adjust the "Xparts" and the "SizeX" for the picture.

The total width of the picture, in pixels becomes: $Xparts * SizeX$.

These are the settings for a 360 degree panorama: $Xparts = 8$, $SizeX = 720$, $lens = 38.6$.

100%, 75%, 50%, 25% (RowBut)

These pre-sets allow you to render smaller pictures. It also affects the size of 'shadow buffers'

Fields (TogBut)

Specifies that two separate *fields* are rendered. Each field is a complete picture. The two fields

Odd (TogBut)

This option indicates that the first field in a video frame begins on the first line.

x (TogBut)

With "Field" rendering, this switches the time difference between the two fields off (0.5 frame).

Border (TogBut)

This allows you to render a small cut-out of the image. Specify a render 'border' with **SHIFT-B** in the 3DWindow (in Camera view of course).

A cut-out is always inserted in a complete picture, including any "BackBuf" that may be present. Set the option "Crop" on to turn this off.

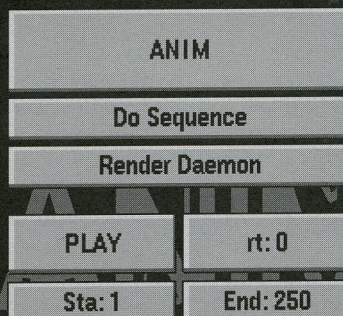
Gamma (TogBut)

Colours cannot be normally added together without consequences, for example when rendering anti-aliasing. This limitation is caused by the way light is displayed by the screen: the colour value 0.4 does not appear half as strong as 0.8 (in actuality it is nearly 0.56!).

This can be solved by assigning the display-hardware an extremely high gamma correction: gamma 2.3 or even higher. This gives a really pale image with 'washed out' dark tints to which dithering must be applied. Blender renders everything internally already gamma-corrected. This produces a more stable anti-aliasing for the eye, i.e. anti-aliasing that does not 'swim'.

To see this difference, render a "Shadeless" white plane with OSA - and with and without "Gamma".

The only time this option should be set to **OFF** is when Blender is used for *image composition*.



ANIM (But)

Start rendering a sequence. This is a 'blocking' process. A square mouse cursor indicates that Blender is busy.

Animations can also be rendered in the 'background'. Use the command line for this: `blender -b file.blend -a`.

Do Sequence (TogBut)

Specifies that the current Sequence strips must be rendered. To prevent memory problems, the pictures of the complete Sequence system are released per rendering, except for the current frame.

→150

RenderDeamon (TogBut)

Indicates to the external network render utility that the current Scene must be rendered (not supported in this version).

Play (But)

This starts an animation playback window. All files from the "Pics" directory are read and played.

→228

rt (NumBut)

For debugging purposes.

Sta, End (NumBut)

The start and end frame of an ANIM rendering.

SizeX: 320

SizeY: 256

AspX: 1

AspY: 1

SizeX, SizeY (NumBut)

The size of the rendering in pixels. The actual value is also determined by the percentage buttons (100%, 75%, etc..)

AspX, AspY (NumBut)

The pixel relationship. The pixels in monitors and video cards are not usually exactly square. These numbers can be used to specify the relative dimension of a pixel.

Targa

Iris

JPEG

Cosmo

HamX

Ftype

Ftype

Movie

Quality: 95

MaxSize: 141

Crop

These buttons specify the graphics file format in which images are saved.

Targa (RowBut)

This is a commonly used, RLE-compressed standard.

Iris (RowBut)

The standard for SGI software.

JPEG (RowBut)

This lossy format can produce strong compression. Use "Quality" to indicate how much compression you want.

Cosmo (RowBut)

Only for SGI: specifies that Cosmo hardware must be used to compress SGI Movies.

HamX (RowBut)

A self-developed 8 bits RLE format. Creates extremely compact files that can be displayed quickly. To be used only for the "Play" option.

Ftype (RowBut)

This switches the Ftype option ON. See the description of the "Ftype" TextBut.

Movie (RowBut)

Only for SGI: Blender writes SGI movies.

Quality (NumBut)

Specifies the quality of the JPEG compression. Also for Movies.

MaxSize (NumBut)

For Movies: the average maximum size of each Movie frame in Kbytes. The compression factor can then vary per frame.

Crop (NumBut)

Specifies that the "Border" rendering must not be inserted in the total image.

For Sequences, this switches the automatic picture *scaling* off. If the pictures are enlarged, the outside edges are cut off.

BW

RGB

RGBA

IRIZ

BW (RowBut)

After rendering the picture is converted to black & white. If possible, the results are saved in an 8 bit file.

RGB (RowBut)

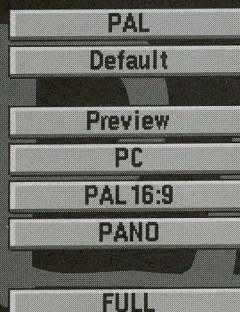
The standard. This provides 24 bit graphics.

RGBA (RowBut)

If possible (not for JPEG) the *alpha* layer is also saved. This provides 32 bit graphics.

IRIZ (RowBut)

Only for Iris format graphics: the Zbuffer is added to the graphics as a 32 bit extra layer.



reference

A number of presets: (In the future, this will be replaced by a user-defined script).

PAL (But)

The European video standard: 720*576 pixels, 54*51 aspect.

Default (But)

Like "**PAL**", but here the render settings are also set.

Preview (But)

For preview rendering: 320*256 pixels.

PC (But)

For standard PC graphics: 640*480 pixels.

PAL 16:9 (But)

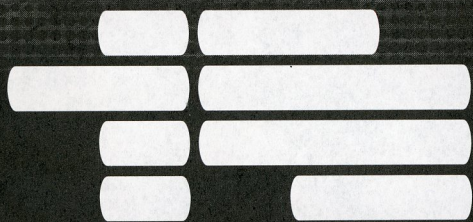
Wide-screen PAL.

PANO (But)

A standard Panorama setting.

FULL (But)

For large screens: 1280*1024 pixels.



Part 10

Appendices

Feature listing

This list also shows information about the current state of development.

These are the codes:

- | | |
|---|---|
| 1 | <i>Only for a limited usage.</i> |
| 2 | <i>Good basis, the provided tools are sufficient.</i> |
| 3 | <i>Well implemented, many tools, but not finished yet.</i> |
| 4 | <i>Implementation and tools are very good, but needs modernisation.</i> |
| 5 | <i>The best there is!</i> |

General

- | | |
|---|--------------------------------|
| 4 | Interface |
| 5 | Data structure and files |
| 3 | Modeling primitives and tools |
| 3 | Animation system |
| 4 | Render engine |
| 3 | Post production editor |
| 1 | Import/export other 3D formats |
| 2 | Plugin system (disabled) |

Windows

- | | |
|---|---|
| 5 | Flexible non-overlapping window layout |
| 5 | 3D window: wireframe/solid/GL-lighted/rendered |
| 4 | In-house buttons and interface toolkit |
| 5 | Combined animation curve and keys window |
| 2 | Schematic diagram window |
| 3 | Sequence editing window for after-effects and video montage |
| 4 | File selecting and file management windows |

Files

- | | |
|---|---|
| 5 | Save all your work in a single file |
| 5 | Files are cross-platform compatible |
| 5 | Other Blender files can be used as libraries (disabled) |
| 4 | read/write Targa, JPEG, Iris, SGI movie, Amiga IFF |
| 1 | read Inventor files |

Light

- | | |
|---|---|
| 5 | Local lights, spotlights, hemispheres, sun. |
| 5 | Textured lights |
| 1 | Spothalo's |
| 4 | Shadow buffer system |
| 5 | Selective lighting for individual objects. |

Render

- | | |
|---|---|
| 4 | Rendering in foreground with direct output |
| 4 | Three rendering layers: for solid faces, transparent faces and halos. |
| 5 | Delta accumulation buffer with jittered sampling for perfect antialiasing. |
| 5 | Any resolution up to 4096x4096 |
| 5 | Field rendering and pre-gamma correction for the best video output possible |
| 5 | Panorama rendering |
| 2 | Plugins for textures and post production (disabled) |

Animation

- 4 | Motion paths: Bezier curves, Nurbs-splines or polygons.
- 5 | Motion curves: XYZ translation/rotation/scaling
- 5 | Motion keys: transformations fixed at a frame
- 3 | Vertex key framing for morphing
- 2 | Inverse kinematics and skeletons
- 5 | Animation curves for light, materials, textures

Modeling

- 4 | A wide range of 3D primitives
- 5 | Deformation Lattices
- 3 | Deformation Skeletons
- 5 | Automatic duplicators

Meshes

- 5 | Triangle and square polygons
- 4 | A wide range of tools: extrude, spin, screw, bend, subdivide, etc.
- 2 | Realtime 'vertex painting' to add vertex colors

Curves

- 4 | Bezier Bspline, polygons
- 5 | Automatic 'hole' detection and triangulation.
- 3 | Automatic beveling; any curve can be a bevel curve
- 4 | A wide range of tools for editing and drawing curves

Surfaces

- 2 | Only Nurbs surfaces
- 2 | Extrude/Spin

Font

- 4 | Adobe type 1 import
- 4 | Type and edit text in 3D

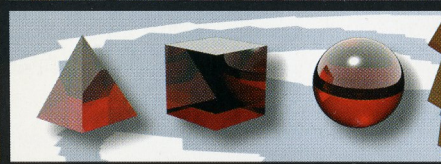
MetaBall

- 1 | Ball and tube primitives

Particles

- 3 | Physics based modeling, for smoke, fire, etc.

PRIMITIVES



TRIANGULAR

CUBIC

SPHERICAL





Command line options

SYNOPSIS

```
blender [-p sx sy w h] [filename]
blender -b|B filename [-S name] [-s nr] [-e nr] [-a] [-f nr]
blender -a image_filename
```

FOREGROUND OPTIONS

Blender can be started without arguments. It then opens a full screen window and reads the file \$HOME/.B.blend. These are the options for a 'foreground' Blender:

- p sx sy w h

Set a custom window size: 'sx' and 'sy' are the lower-left coordinates, 'w' and 'h' are width and height respectively. Example: blender -p 280 224 720 576.

filename

With a blenderfile as argument, Blender reads this file at startup.

BACKGROUND OPTIONS

Blender can run without window as well, for 'background' rendering:

- b filename

Start a Blender in background mode and low priority. Read the file.

- B filename

Start up a Blender in background mode and normal priority. Read the file.

- S name

Set Scene 'name'

- s nr

Set start frame at 'nr'

- e nr

Set end frame at 'nr'

- a

Start rendering the animation. Save images or movie as indicated in the Blenderfile.

- f nr

Start rendering of a single frame 'nr' Save the image as indicated in the Blenderfile.

Examples:

```
blender -b /usr/data/rt.blend -a
```

(Render and save all frames from start to endframe)

```
blender -b /usr/data/rt.blend -S logo -s 11 -e 20 -a
```

(Render and save frames 11 to 20 from Scene 'logo')

```
blender -b /usr/data/rt.blend -f 1 -f 38 -f 89
```

(Render and save the frames 1 38 and 89)

PLAYBACK OPTION

- a image_filename

To allow you to view sequential numbered images, Blender has a simple, but efficient animation playback option. Blender first reads all the files in memory and then displays them as a flip book. Check in advance to make sure sufficient memory is available.

Glossary of Blender terms

active

Blender makes a distinction between ~selected and ~active. Only one Object or item can be ~active at any given time, for example to allow visualisation of data in buttons.

AKEY, BKEY, ...

The keyboard button 'A' 'B'

alpha

The alpha value in an image denotes opacity, used for blending and antialiasing.

anti-aliasing

Algorithm designed to reduce the stair-stepping artifacts that result from drawing graphic primitives on a raster grid.

back-buffer

Blender uses two buffers to draw the interface in. This double-buffering system allows one buffer to be displayed, while drawing occurs on the back-buffer. For some applications in Blender the back-buffer is used to store colour-coded selection information.

channel

1 Some DataBlocks can be linked to a series of other DataBlocks. For example, a Material has eight ~channels to link Textures to.

2. Each Ipo block has a fixed number of available ~channels. These have a name (LocX, SizeZ, enz.) which indicates how they can be applied. When you add an IpoCurve to a channel, animation starts up immediately.

clipping

Removing, before drawing occurs, of vertices and faces which are outside the field of view.

Child

Objects can be linked to each other in hierarchical groups. The Parent Object in such groups passes its transformations through to the Child Objects.

DataBlock (or 'block')

The general name for an element in Blender's Object Oriented System.

double-buffer

Blender uses two buffers (images) to draw the interface in. The contents of one buffer is displayed, while drawing occurs on the other buffer. When drawing is complete, the buffers are switched.

extend select

Add new selected items to the current selection. 1

face

The triangle and square polygons that form the basis for Meshes or for rendering.

flag

A programming term for a variable which indicates a certain status.

Gouraud shading

The colours of vertices are linearly interpolated across the face to achieve smooth gradual colour transitions.

hierarchy

Objects can be linked to each other in hierarchical groups. The Parent Object in such groups passes its transformations through to the Child Objects.

Ipo

The the main animation curve system. Ipo blocks can be used by Objects for movement but also by Materials for animated colours.

IpoCurve

The Ipo animation curve.

item

The general name for a selectable element, e.g. Objects, vertices or curves.

layer

A visibility flag for Objects, Scenes and 3Dwindows. This is an extremely efficient method for testing Object visibility.

link

The reference from one DataBlock to another. In fact, it is a 'pointer' in programming terminology.

local

1 Each Object in Blender defines a ~local 3D space, bounded by its location, rotation and size. Objects themselves reside in the ~global 3D space.

2. A DataBlock is ~local, when it is read from the current Blender file. Non-local blocks (library blocks) are linked parts from other Blender files. (Disabled in V1.5)

mapping

The relationship between a Material and a Texture is called the 'mapping'. This relationship is two-sided. First, the information that is passed on to the Texture must be specified. Then the effect of the Texture on the Material is specified.

ObData block

The first and most important DataBlock linked by an Object. This block defines the Object ~type, e.g. Mesh or Curve or Lamp.

Object

The basic 3D information block. It contains a position, rotation, size and transformation matrices. It can be linked to other Objects for hierarchies or deformation. Objects can be 'empty' (just an axis) or have a link to ObData, the actual 3D information: Mesh, Curve, Lattice, Lamp, etc.

selected

Blender makes a distinction between ~selected and ~active. Any number of Objects can be ~selected at once. Almost all key commands have an effect on ~selected Objects.

Single User

DataBlocks with only 1 user

smoothing

A rendering method that performs vertex-normal interpolation across a face before lighting calculations start. The individual facets then are not visible anymore.

transform

Change a location, rotation or size. Usually applied to Objects or vertices.

user

When a DataBlock is referenced by another DataBlock, it has a user.

vertex (vertices)

The general name for a 3D point. Besides an X,Y,Z coordinate, a vertex can have colour a normal vector and a secection flag.

Zbuffer

For a Zbuffer image, each pixel is associated a Z-value, derived from the distance in 'eye space' from the Camera. Before each pixel of a polygon is drawn, the existing Zbuffer value is compared to the Z-value of the polygon at that point.

It is a common and fast visible-surface algorithm.

Index

3DWindow	
header	185
hotkeys	189
mouse.	187
window	187
3D	
view manipulation.	33
view pre-sets.	34, 186
view modes.	34, 186

A

Add	
Objects.	34
Materials	274
Scenes	174
Screens	174
Sequence strips.	217
activating	25, 88
alpha.	148
ambient light	263
animation playback window	228, 291
AnimButtons	.. 266
anti-aliasing	
graphics pre-process.	159, 254
rendering	147 289
append files	180
ascii files	164
aspect ratio (rendering)	292
assign Materials	
Curve	.. 103
Mesh	96
Surface	103
Auto save	175
background image ..	156, 231 288
beveling	
Curves (BevObject)	102, 282
pre-sets	101 282

B

Bezier	
3D curve	.98
handles	98
lpoCurve	117
blend texture	258

bluescreen video	159
border	
selection	190
render	190, 291
bump texture	253, 258
Build Effect ..	269
ButtonsWindow	
header	205
hotkeys	206
types	205
button types..	79

C

Camera	
block ..	.65, 111b
EditButtons	.. 286
cartoon animation and 3D	.. 159
channel	
lpo	113
Texture	.. 134
Child	.89
clouds texture	256
ColorBand	.. 252
colormap files (FType)	.. 150, 288
convert menu	.. 191
copy menu	191
Curve	
animation curves ..	.. 113
block.	.. 65, 97
EditButtons	281
EditMode hotkeys.	201
handles	.. 98, 201
modeling	.. 101
motion path	126, 268
vertex keys	.. 125
cross effect ..	.. 153
cyclic	
animation	117
curves.	101

D

DataBlocks	64
DataBrowse	181
data structure	
example	.. 20
general	63
DataView	181
DisplayButtons.	.. 288
double-sided	280
DrawKey option	119

Draw Type	
3D window (mode)	.34
Objects.	.273
duplicate command	191
duplication presets	177
Duplicator Objects.	129, 267

E	
EditButtons	273
EditMode	
hotkeys	197
Objects 26	
Mesh	.. .92
effects	
Build	.. .269
Particles.	270
effector (lka) 106	
Euler angles	.. .118
explosion Particles	131
exposure	.263
extend mode lpoCurve	117
extrude	
Mesh	.. .92, 276
Curve	.. .201
Surface	201
environment variable	14

F	
faces	
modeling	.92
rendering.	144
fields.	.. .253, 290, 291
files	
Blender format	.. .72
graphics formats	150, 292
Inventor	.. .163
saving and loading	29, 178
versions	176
Videoscape	164
FileManager	181
FileSelect	.. .179
FileWindow	
header	178
window	178
fill polygons	198
fire particles.	131
fly mode	191
Font	
block.	104
EditButtons	283

editing	104
EditMode hotkeys.	.202
fractal subdivide.	.280

G	
gamma correction	.291
gestures	.88
Grabbing	88, 192
graphics file formats	150, 292
grid	
menu	.231
snapmenu	195

H	
halo buttons	.. .248
handles	.. .98, 201
hemi lamp.	139
hierarchy	.. .89
HSV option.	239

I	
lka	
animation	130
block.	.65, 106
EditButtons	285
EditMode hotkeys.	203
modeling	106
Image	
block.	66
load	254
Texture	.. .253
image aspect (rendering)	.292
ImageWindow	224
ImageSelectWindow	226
import 3D files	.. .163
Insert Key	114, 123, 192
Installation	14
FreeBSD.	.. .17
Linux.	17
SGI	14
Inventor files	163
lpo	
block.	.66, 113
inserting	114, 192
lpoCurve	
editing.	117 118, 124
types	118, 212
lpoKey	119, 207

IpoWindow	
editing in	114, 209
header	207
hotkeys	114, 201
window	209
InfoWindow	
header	174
UserMenu	175

J

join 192

K

Key	
block.	.66, 123
insert.	192
IpoKey	119
Object	113
slurping	125
VertexKey	123, 124
key alpha	148
keyboard	
standard usage	14
knots (Nurbs)	...98, 281

L

Lamp	
block. ..	.65
Ipo	207
types	139, 233, 234
textures	140, 235
tips ..	140
LampButtons.	233
Lattice	
block.	.65, 105
deformation	90
EditButtons	285
modeling	105
vertex keys 125	
layers.	.34, 185, 193
lens	
3DWindow	231
Camera	.. 286
lens flares	248, 249, 250
loading files	.29, 178
load Font	.283
Library	
block.	66

files	180
structure	74
light calculation	133
lighting	140
limb	106
link 64, 71	
link menu	193
LocalView	185

M

magic texture	257
mapping	134, 243
marble texture.	257
Material	
animation ..	138
block.	.65
examples 134	
indices	.274
Ipos	138, 207
making links ..	70
mapping	134
MaterialButtons	238
Mesa ..	14
Meta sequence.	153
Mesh	
block.	.65, 91
EditButtons	275
EditMode hotkeys. ..	.. 198
modeling	.36, 92
vertex keys	123
MetaBall	
block. ..	.65, 111
EditButtons	284
modeling ..	111a
memory usage	.. 149
mip-mapping	.. 253
mist	263
motion capture (recording mouse).	211
mouse	
standard usage	.24
window isage	171

N

noise texture	.259
Normals	
flip ..	.280
recalculation	.201
rendering.	144
Number menu	84
numerical pad	188

Nurbs
curve .. .98



ObData block. .66
Object
block... 64, 87
Duplicator 129, 267
lpo 207
object oriented system 19
OopsWindow
header .220
hotkeys 222
mouse 221
window .. 221
output (images) .. 288



pad (keyboard) 188
panorama rendering 290
Parent (parenting). .89, 194, 203
Particles
buttons .. 270
system 130
Perlin noise .. 256
poly curve 101
premul alpha 148
PupMenu .. 84



recording mode 211
render
anti-aliasing .. 289
faces and normals .. 144
limits. 149
principles. 133, 144, 147
resolution 292
shadows 290
window location 280
resolution 292
rotation
centre 186
limits (Euler) 118
mode 88, 194
rotoscoping 156



saving files .29, 178
scaling 88, 195
Scene
block. 64
browse 174
set (backdrop Scene) .289
Screens
block. .64
browse 174
hotkeys 171
structure ... 23, 78
screw (Mesh tool) .276
selection .. 25, 88
separate (Objects tool). 197
Sequence
editor 150
effects .. 153
header .. 214
hotkeys .. 217
mouse. 217
rendering 150, 291
window 214
Single User 133
shading 139, 235
shadow buffers. 290
shadow rendering 197
shear 197
Skeleton
animation 130
buttons 286
deformation .90, 110
modeling 106, 203
sky
options 260
rendering. 148
slurphing 125, 267
smoke particles .. 131
smoothing .92, 144, 275
snap menu. 195
spin
Curve .283
Mesh 95, 279
Surface .. 283
split Mesh. .. 279
spot lamp 139
spring modeling 276
stars. 263
sticky texture .242, 275
stucci texture 258
subdivide
Curve .. 279
Mesh .279
Surface 279

sun lamp	139
Surface	
block.	102
EditButtons	281
EditMode hotkeys.	202
modeling	103
vertex keys	125

T

Text on Curve	283
Texture	
animation	138, 255
block.	66
mapping	134
Mesh	275
Particle movement.	131
solid (Mesh tool 'Noise')	280
types	137
TextureButtons	251
time offset	267
ToolBox	83, 90
track, tracking	126, 129, 196, 266
translation	88, 192
triangulation.	198

U

undo	73
UserMenu	175
users	71

V

view (3D)	
manipulation	33
modes	34, 186
pre-sets.	34
ViewButtons	231
versions (files)	176
vertex	
colour	155
Curve	97
Duplicator	90
Mesh	91
Parent	89, 90, 197
Surface	102
VertexKeys	123, 125, 207
VertexPaint	
options.	155, 204, 275

buttons	287
VRML files	163

W

window	
animation playback.	228
hotkeys	171
types	23, 78, 174
wood texture	256
World	
block 64	
lpo	207
WorldButtons	260

Z

Zbuffer	147
---------	-----

riffraff

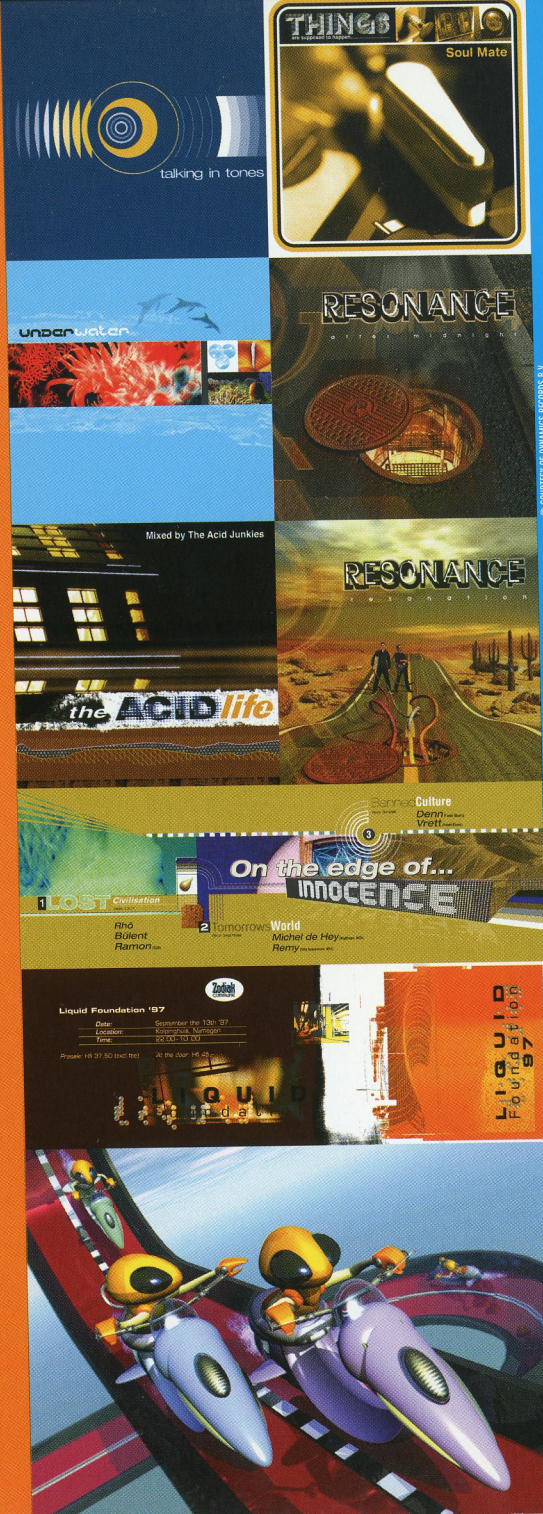
2D3Design

Riff Raff are:

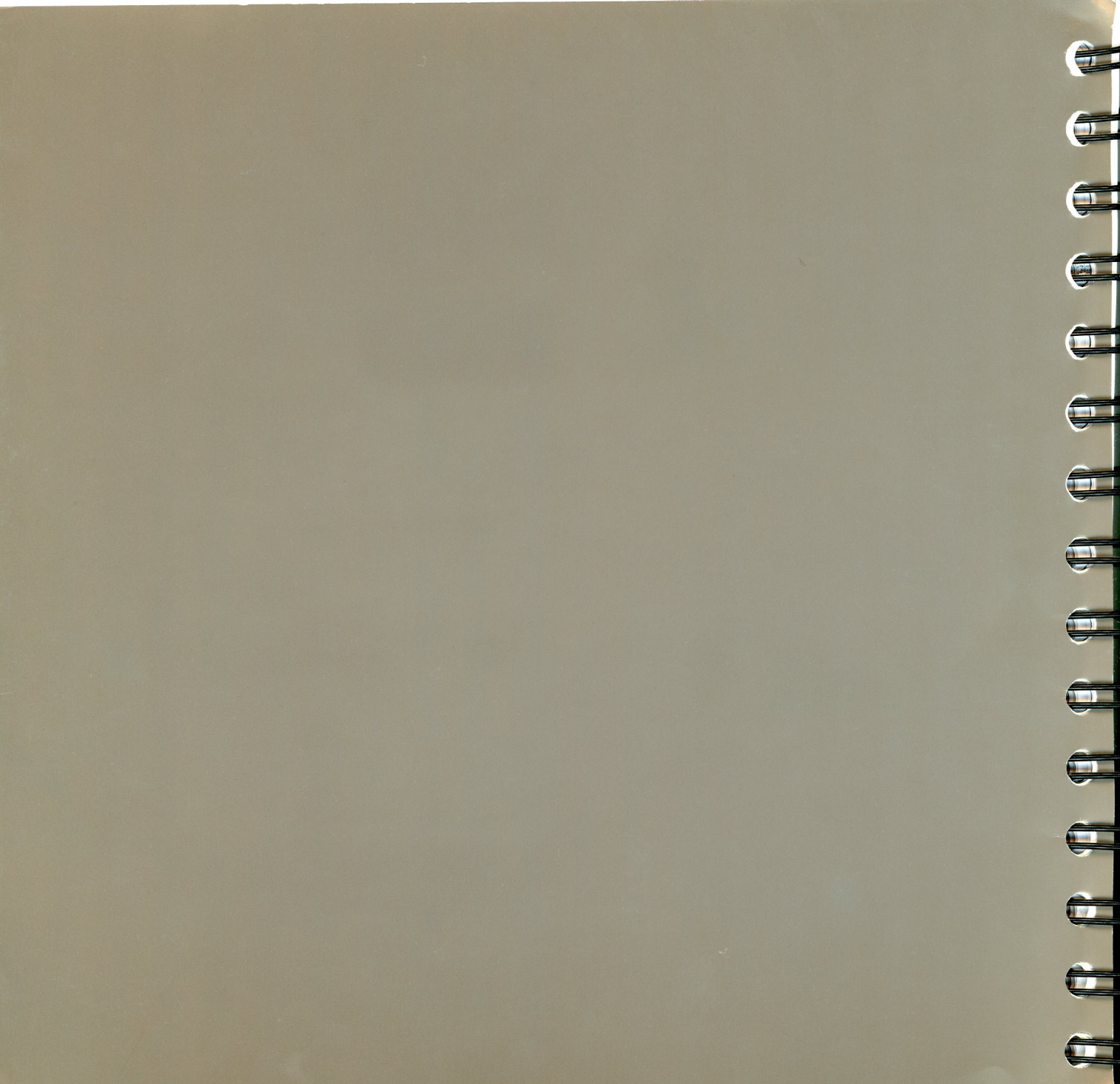
Bas Waijers
Jurgen van Zachten

P.O. Box 11481
1001 GL Amsterdam
The Netherlands

+31 (0) 20 689 10 36

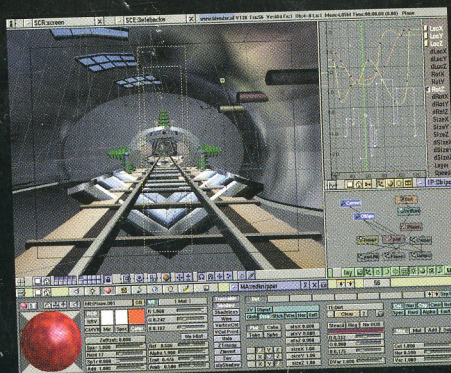


© COURTESY OF DYNAMICS RECORDS S.V.



Blender

V1.5 manual



Free software for SGI and
Linux/ FreeBSD i386 PC's

<http://www.blender.nl>

Complete 3D modeling
and animation suite